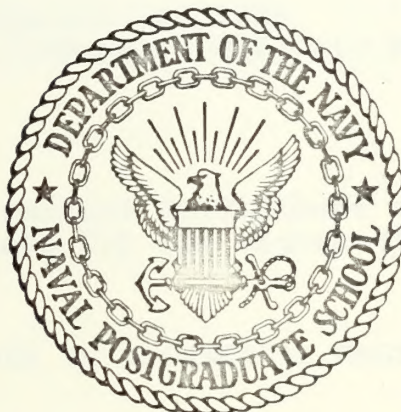


A GENERAL PURPOSE THREE DIMENSIONAL
STRESS ANALYSIS PROGRAM

Emmanuel Leonidas

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

A GENERAL PURPOSE THREE DIMENSIONAL
STRESS ANALYSIS PROGRAM

by

Emmanuel Leonidas

Thesis Advisor:

G. Cantin

December 1971

Approved for public release; distribution unlimited.

A General Purpose Three Dimensional
Stress Analysis Program

by

Emmanuel Leonidas
Lieutenant Commander, Hellenic Navy

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1971

ABSTRACT

A computerized solution for three dimensional stress analysis was developed. The solution was based on the finite element technique employing twenty and thirty-two nodal point three dimensional isoparametric elements. The solution is applicable to linear elastic structures composed of homogeneous and isotropic materials subjected to static loading under constant temperature. Deformed states are restricted to small displacements and displacement gradients. The program is coded in standard FORTRAN IV and is machine independent except for one subroutine which may be adjusted to specific installations. Independence of core space requirements is achieved by use of direct access storage facilities.

TABLE OF CONTENTS

I.	INTRODUCTION	-----	6
II.	NATURE AND FORMULATION OF THE PROBLEM	-----	8
	A.	FINITE ELEMENTS	----- 10
	B.	ELEMENT AND TOTAL STIFFNESS MATRICES - CONSISTENT LOADS	----- 13
	C.	STRESSES AND STRAINS	----- 14
III.	DESCRIPTION OF THE COMPUTER PROGRAM	-----	15
	A.	CONTROL PROGRAM	----- 17
	B.	SUBROUTINE MAINE	----- 17
	1.	Reading of Input Data	----- 18
	2.	Formation of Element Nodal Coordinate Arrays	----- 19
	3.	Formation of Element Stiffness Matrices -	20
	4.	Formation of Total Stiffness Matrix	---- 21
	5.	Formation of Load Vector and Application of Boundary Conditions	----- 21
	6.	Solution of the System - Determination of Reactions	----- 24
	7.	Evaluation of Nodal Strains and Stresses -	25
	C.	SUBROUTINE DISK	----- 26
IV.	PROBLEM AND INPUT DATA PREPARATION	-----	29
	A.	STRUCTURE MODELING	----- 29
	B.	INPUT DATA PREPARATION	----- 30
	1.	Connectivity Matrix	----- 30
	2.	Nodal Point Coordinates	----- 31

3. Material Properties -----	32
D. DIRECT ACCESS FILE REQUIREMENTS -----	38
V. SOLUTION OF A SIMPLE PROBLEM -----	39
A. ALTERNATE FORMULATION -----	40
B. DISCUSSION OF RESULTS -----	41
APPENDIX A. Computer Program Variables -----	48
A. DATA FILES CHARACTERISTICS -----	48
B. VARIABLES APPEARING IN COMMON STATEMENTS ---	48
C. OTHER IMPORTANT VARIABLES -----	53
APPENDIX B. Direct Access Data Files Requirements ---	56
A. DATA FILE SPACE REQUIREMENTS -----	56
B. CONTROL CARDS -----	57
C. DEFINE FILE STATEMENTS -----	59
APPENDIX C. Sample Input Data Decks for TRISOP -----	60
A. QUADRATIC ELEMENTS -----	60
B. CUBIC ELEMENTS -----	62
APPENDIX D. Sample Output for TRISOP -----	64
APPENDIX E. Sample Output for JCL -----	73
APPENDIX F: Deck Structure for Computer Programs in IBM 360 OS -----	76
A. TRISOP - QUADRATIC ELEMENTS -----	76
B. TRISOP - CUBIC ELEMENTS -----	77
C. AUXILIARY PROGRAM JCL -----	78
APPENDIX G. Computer Listings for TRISOP -----	79
A. QUADRATIC ELEMENTS -----	81
B. CUBIC ELEMENTS -----	138
APPENDIX H. Listing for Auxiliary Program JCL -----	153

APPENDIX I. Listings of Integration Subroutines	-----	163
A. QUADRATIC ELEMENTS - TWO GAUSS POINTS	-----	163
B. QUADRATIC ELEMENTS - THREE GAUSS POINTS	-----	164
C. QUADRATIC ELEMENTS - FOUR GAUSS POINTS	-----	165
D. CUBIC ELEMENTS - THREE GAUSS POINTS	-----	166
E. CUBIC ELEMENTS - FOUR GAUSS POINTS	-----	167
F. CUBIC ELEMENTS - FIVE GAUSS POINTS	-----	168
LIST OF REFERENCES	-----	169
INITIAL DISTRIBUTION LIST	-----	170
FORM DD 1473	-----	172

I. INTRODUCTION

The finite element technique is a relatively new method for stress analysis of structural continua.

Due to its almost limitless abilities this technique has recently been extensively used in structural analysis and considerable research has been conducted toward the development of elements and techniques satisfying the requirements of advanced structural technology.

The present work concerns the development of a general purpose digital computer program for three dimensional stress analysis, named TRISOP, and employing the finite element method.

The program developed, in its present form, is applicable to the solution of problems concerning linearly elastic structures, composed of homogeneous and isotropic materials, and subjected to static loading at constant temperature. The deformed states are restricted to small displacements and displacement gradients. The use of an equation solver of, in principle, infinite capacity devised by Cantin [Ref. 2] in conjunction with direct access storage devices enabled the memory space requirements to be independent of the size of the problem, the latter being limited only by the execution time and availability of storage devices.

At the time of initiation of the development of TRISOP the most promising finite elements, in the three dimensional

analysis field, were the family of isoparametric elements. Two such elements, namely the 20 (Quadratic) and 32 (Cubic) nodal points elements, are used in the program.

The present form of TRISOP is an improved version of a program coded by Professor Gilles Cantin of the Naval Post-graduate School and employing quadratic isoparametric finite elements. The major improvements brought to this initial form of TRISOP by the present version are, the extension to cubic elements, inclusion of predefined displacements in the boundary conditions, improved execution time and independence from the size of the problem.

The structure of the program follows a modular pattern in order to provide for further improvements and, possibly, adjustment to new developments in the field.

II. NATURE AND FORMULATION OF THE PROBLEM

The problem considered is a three dimensional stress analysis of a statically loaded structure under constant temperature. The analysis is restricted to deformed states within the elastic region and involving small displacement gradients. Consequently the linear strain tensor only need be considered and the strain-displacement relations can be expressed as:

$$\begin{Bmatrix} \epsilon_{xx} \\ \epsilon_{yy} \\ \epsilon_{zz} \\ \epsilon_{xy} \\ \epsilon_{xz} \\ \epsilon_{yz} \end{Bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \end{bmatrix} \begin{Bmatrix} u \\ v \\ w \end{Bmatrix} \quad (2.1)$$

or in more compact form:

$$\{\epsilon\} = [D]\{u\} \quad (2.2)$$

Where both $\{\epsilon\}$ and $\{u\}$ are point functions.

As a consequence of the above the principle of superposition applies to strains and displacements. The stress-strain law is assumed to be linear and thus the principle of superposition applies for stresses as well. Therefore,

displacements, strains and stresses of two distinct deformed states are algebraically additive resulting into a new compatible deformed state. Consequently initial stresses and strains need not be considered in the formulation of the problem. If such strains and stresses exist they will have to be superposed to the results of the solution which will be relative to a reference state, namely the state of the structure before the application of the load considered in the problem.

In addition to small displacement gradients, the analysis is restricted to displacements small enough that equilibrium and compatibility relations can be referred to the undeformed geometry of the structure (here and in what follows by undeformed state is meant the reference state).

Due to the restriction of small displacements the question of stability does not enter the analysis [Ref. 7]. The structure may be composed of more than one materials which are assumed to be homogeneous and isotropic. The constitutive laws for this problem can be expressed in matrix form as:

$$\begin{Bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \sigma_{xy} \\ \sigma_{xz} \\ \sigma_{yz} \end{Bmatrix} = \frac{E^*}{(1+\mu)(1-2\mu)} \begin{bmatrix} 1-\mu & \mu & & & & \\ \mu & 1-\mu & & & & \\ \mu & \mu & & & & \\ & & \frac{1-2\mu}{2} & & & \\ & & & \frac{1-2\mu}{2} & & \\ & & & & \frac{1-2\mu}{2} & \end{bmatrix} \begin{Bmatrix} \epsilon_{xx} \\ \epsilon_{yy} \\ \epsilon_{zz} \\ \epsilon_{xy} \\ \epsilon_{xz} \\ \epsilon_{yz} \end{Bmatrix}$$

or in more compact form:

$$\{\sigma\} = [E]\{\epsilon\} \quad (2.3)$$

Where $\{\sigma\}$ is a point function, E^* is Young's modulus and μ is Poisson's ratio.

A. FINITE ELEMENTS

The main feature of the finite element technique is the approximation of the elastic continuum possessing an infinite number of degrees of freedom by an assemblage of substructures, the finite elements, each possessing a finite number of degrees of freedom.

Among the various types, two elements of the isoparametric family were chosen for this development; namely, the 20 and 32 nodal point three dimensional elements. The concepts underlying the construction and use of these elements as well as the resulting properties will not be presented here except for those necessary for the presentation of the structure of the computer program. An extensive treatment of this family of finite elements as well as the isoparametric concept may be found in Refs. 4 and 8.

Two systems of reference are used throughout. First, a "local" non dimensional and, in general, non cartesian system (ξ, η, ζ) is used for convenience of computation of the element stiffness matrices. Then everything is transferred in a "global" cartesian system of reference (x, y, z) for the final computations of the assembled system.

A numbering convention for the element nodes is employed and is shown in Figures 1 and 2. A different convention is used for numbering the nodes of the assembled structure and is described in Section IV.A.

By use of shape functions $N_i(\xi, \eta, \zeta)$, describing the variation of element properties in the local system, every element property $\phi(\xi, \eta, \zeta)$ can be expressed in terms of its nodal values as:

$$\phi(\xi, \eta, \zeta) = \langle N_i \rangle \{ \phi_i \}$$

Thus the displacements of points within an element will be given by:

$$\begin{Bmatrix} u(x,y,z) \\ v(x,y,z) \\ w(x,y,z) \end{Bmatrix} = \begin{bmatrix} N_1 & 0 & 0 & N_2 & 0 & 0 & \dots \\ 0 & N_1 & 0 & 0 & N_2 & 0 & \dots \\ 0 & 0 & N_1 & 0 & 0 & N_2 & \dots \end{bmatrix} \begin{Bmatrix} u_1 \\ v_1 \\ w_1 \\ u_2 \\ v_2 \\ w_2 \\ \vdots \end{Bmatrix}$$

or in more compact form:

$$\{u\} = [N]\{u_i\}$$

An extensive treatment of the shape function concept may be found in Ref. 8.

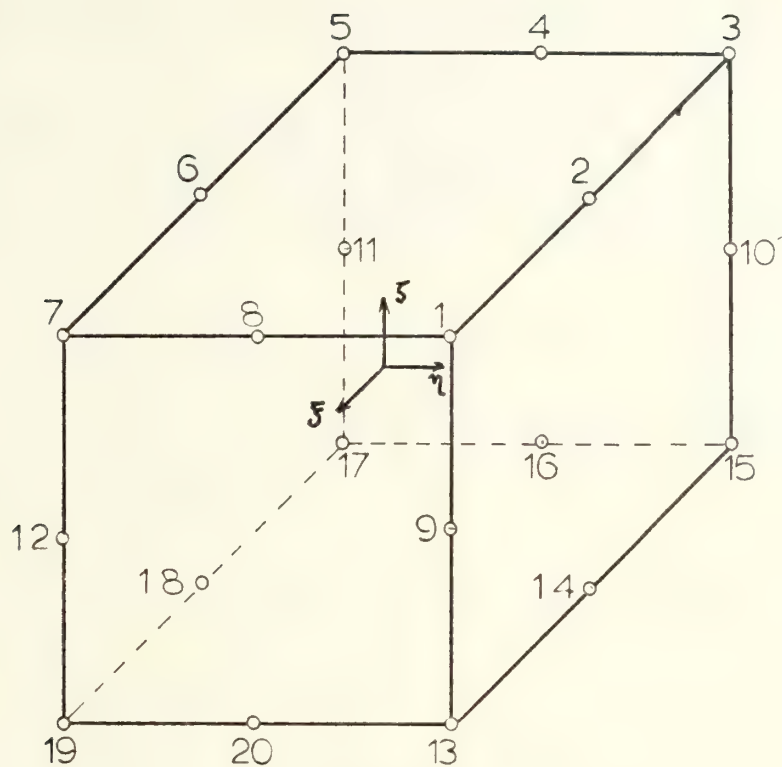


Figure 1.

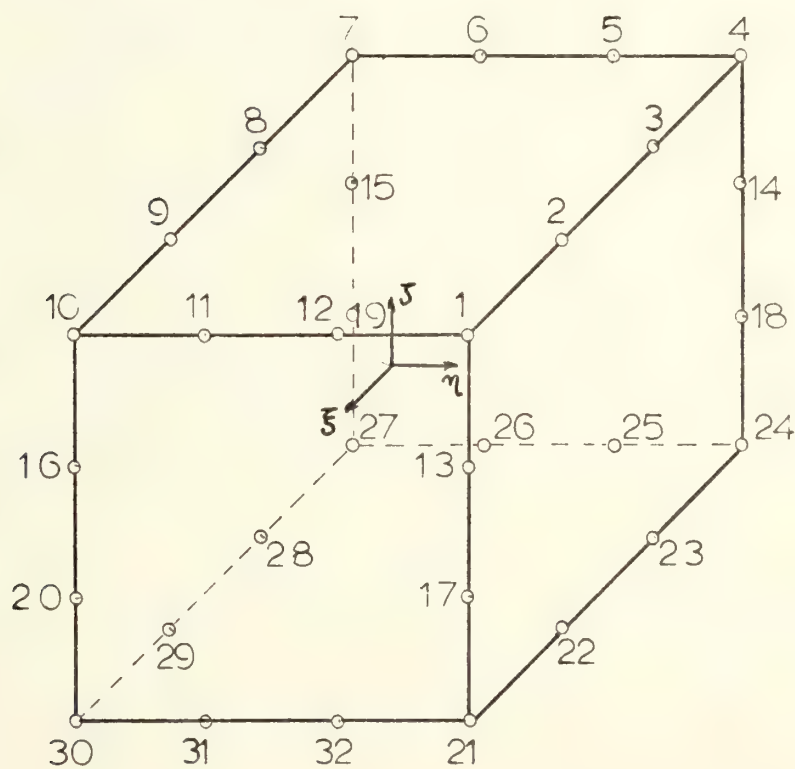


Figure 2.

The shape functions used in this development belong to the Serendipity family and are shown in Refs. 4 and 8.

B. ELEMENT AND TOTAL STIFFNESS MATRICES - CONSISTENT LOADS

Every individual finite element can be visualized as an independent structure and thus the derivation of a stiffness matrix is possible.

The derivation of the element stiffness matrix $[K_e]$ is shown in Ref. 1 where the following integral expression is finally obtained:

$$[K_e] = \int_{-1}^1 \int_{-1}^1 \int_{-1}^1 [N^*]^T [E] [N^*] \det[J] d\xi d\eta d\zeta \quad (2.4)$$

where: $[N^*] = [D][N]$

and $[J]$ is the Jacobian matrix of the transformation.

This expression for the element stiffness matrix is ideally suited for Gaussian numerical integration. Computational considerations on the performance of this integration as well as the choice of integration points are presented in Ref. 1.

After formation of the stiffness matrix for each element, the total stiffness matrix $[K]$ for the structure can be formed by superposition of corresponding stiffness coefficients.

The force-displacement relations for the whole structure can then be expressed as:

$$\{F\} = [K]\{u\} \quad (2.5)$$

where $\{u\}$ is a vector containing all nodal displacements of the structure and $\{F\}$ a vector of consistent concentrated loads applied at the nodes and equivalent to the loading applied to

the structure. The determination of the consistent loads is not incorporated in the present solution and it is assumed that it is known.

Relations (2.5) represent a system of linear equations. When this system is initially formed, both $\{F\}$ and $\{u\}$ will in general contain known and unknown elements. By further manipulation the system can be brought to the standard form with all unknowns on one side and its solution will result into the determination of all unknown nodal displacements and forces (see Section III.B.5).

C. STRESSES AND STRAINS

By use of the strain-displacement relations (2.2) and the displacements obtained from the solution of system (2.5) the vectors of nodal strains can be determined.

However the application of relations (2.2) is only possible at the element level in the finite element theory. Thus the nodal strains for each element separately are first evaluated.

Further by applying the stress-strain relations (2.3) the element nodal strains are evaluated.

Except for nodes lying on the interfaces between elements of different materials, a stress analysis requires the determination of average nodal strains and stresses of all elements sharing common nodes. This is accomplished by simply averaging the algebraic values of the stress and strain components at each node.

III. DESCRIPTION OF THE COMPUTER PROGRAM

Computerized solutions to structural problems based on the finite element technique have, in the recent years, led to a considerable amount of research in three disciplines: structural analysis, numerical analysis and computer programming. Thus new elements and computational techniques are continuously being developed paralleled by the development of computing equipment. For this reason it is not unusual for such computer programs to become obsolete within a few years of their completion.

Under these circumstances, the development of a successful program is required to follow rules which will preserve its effectiveness as long as possible by an adjustment process. These rules are summarized to:

- a. Standard coding and machine independence enabling the program to be used by various installations regardless of the computing equipment available, and
- b. Modular construction allowing adjustment to new developments without requiring the construction of the entire program.

These principles have governed the development of TRISOP at all stages that brought it to its present form.

The structure of TRISOP follows closely the analytic development of Section II. (Also see Flow Diagram.)

The program is coded in standard FORTRAN and is machine independent except for the subroutine DISK via which transfer of data to and from direct access storage devices is accomplished. All calculations are performed in double precision in order to provide adequate accuracy in spite of the round off errors.

The memory space requirements are independent of the size of the problem, the latter being limited only by the size of direct access storage space available and execution time.

The extension of the capabilities of the program, as for example inclusion of thermal stress analysis and nonhomogeneous materials will easily be accomplished without interfering with parts of the program that are not altered by these changes.

A number of checking points was incorporated in order to assure the proper development of the execution. Whenever required, error and/or warning messages are given in the output. If the continuation of the solution is useless due to the error detected the execution for this problem is stopped and the solution of the next problem (if such exists) begins.

The most important variables used throughout the program are shown in Appendix A and a brief description of their function is given.

The basic characteristics of the two programs, for 20 and 32 nodal point elements, are the same. The minor existing differences are discussed in Appendix G where listings of both programs are given.

In what follows the function of the different modules of the program is explained with reference to the analytic development of the solution as it is presented in Section II.

A. CONTROL PROGRAM

The control program serves no other purpose but to initiate the calling sequence of the different modules of the program and read, from input data cards, the characteristics (number and length of records) of the data files as these are requested by the JCL¹ cards. These characteristics are later on (in subroutine INPUT) checked for correctness and adequacy.

B. SUBROUTINE MAINE

Subroutine MAINE is the actual main program. It is coded in subroutine form in order to provide for the solution of more than one problem in one run by simply restarting the calling sequence of the different modules. Its function consists of a step by step calling procedure. The only numerical operation executed by subroutine MAINE itself is the computation of the average nodal stresses and strains after the element nodal stresses and strains have been computed by subroutine STRESS. In the coding of subroutine MAINE a timing procedure has been incorporated giving at the output the execution time for each step. This is an optional feature not connected with the execution of the problem itself and can be removed if required. However, it provides useful data for studying the factors affecting the execution time of a problem.

¹ JCL = Job Control Language

The calling procedure and the function of the subroutines called are as follows:

1. Reading of Input Data

By calling subroutine INPUT all input data cards are read. First, the general characteristics of the problem (as number of elements, number of nodal points, etc.) are read and used for the computation of other characteristics (as total number of equations, number of blocks per column). All general characteristics of the problem are stored in a common storage area and used throughout the program by most of the subroutines. Subsequently the arrays of connectivity, nodal point coordinates, material properties, concentrated load data, and boundary conditions are read. All arrays except for the material properties (which is of pre-specified size) are stored in data files on direct access storage devices. This feature enables the program to be independent of the size of the problem. The bandwidth and the number of blocks per row of the total stiffness matrix are computed from connectivity data. All input data as well as those computed are printed out for checking purposes.

The nature of the program allows the solution of a great variety of problems. For this reason and in order to avoid limitations on the form of the input data most reading formats are not specified but left to the choice of the user by use of object time formats.

Within subroutine INPUT several checks are made in order to insure the consistency of some data. All such checks

throughout the program are accomplished by calling the proper entry of subroutine ERROR. The checks made by subroutining INPUT are the following.

a. Stop Trap for the Last Problem

The last problem of each run is followed by an indicator card, as explained in Section IV. Once this indicator card is read, the execution is stopped.

b. Number of Different Materials

The maximum permissible number of different materials present in the structure to be analyzed is ten. If a problem involves a number of materials greater than ten, the execution is stopped and an error message is given in the output.

c. Characteristics of Data Files

The characteristics of data files, namely the number and size of records for each file, are given to the program once for every run in the main program but their consistency is checked for every problem of the run. If the size of one or more of these characteristics is smaller than that required for the problem the execution is stopped for this problem and execution of the next problem (if such exists) begins. In case the size of one of the data files characteristics exceeds that required for the problem, a warning message is given at the output and execution continues.

2. Formation of Element Nodal Coordinate Arrays

The coordinates of nodal points are provided to the program via subroutine INPUT in a sequence following the node

numbering convention for the whole structure. However, at various points of the program the use of arrays containing the nodal coordinates of each individual element is required. The formation of such arrays is accomplished by subroutine SORT. The sequence of elements of these arrays follows the numbering convention of the element nodes. Each array formed is stored in a direct access data file.

3. Formation of Element Stiffness Matrices

The individual element stiffness matrices $[K_e]$ are formed by subroutine FSTF in the following way.

The element stiffness matrix is given by the expression

$$[K_e] = \int_{-1}^1 \int_{-1}^1 \int_{-1}^1 [N^*]^T [E] [N^*] \det[J] d\xi d\eta d\zeta \quad (3.1)$$

First the matrix of elastic constants $[E]$ is computed by calling subroutine ELPROP. In case the whole structure is made of one material this operation is executed only once since all elements have a common elastic matrix. In the case of more than one material matrix $[E]$ is computed every time the element considered is of different material than the previous one. Subsequently subroutine CUB is called to perform Gaussian numerical integration. For each Gauss point subroutine FORMK is called by CUB and the integrant of expression (3.1) is formed in a step by step procedure and evaluated at the integration point. Once an element stiffness matrix is formed it is stored on disk.

4. Formation of Total Stiffness Matrix

The formation of the total stiffness matrix is performed in subroutine MERGE by superposition of the corresponding stiffness coefficients. In order to avoid the presence of the whole stiffness matrix in core, it is partitioned in blocks of pre-specified size. These blocks are formed one at a time by selecting the proper elements from the element stiffness matrices. Each element stiffness matrix is scanned in order to detect elements belonging to the block under formation. Every completed block of the total stiffness matrix is in turn stored on disk. The method of partitioning and storing of the total stiffness matrix is established in Ref. 2. A thorough study of this paper is essential in order to comprehend the procedure outlined here as well as to become familiar with the program itself.

5. Formation of Load Vector and Application of Boundary Conditions

After the formation of the total stiffness matrix the completion of system 2.5 requires the formation of vectors $\{F\}$ and $\{u\}$ and some further manipulation of the system in order to bring it in the standard form.

The load vector $\{F\}$ is partitioned into a number of smaller vectors the size of which corresponds to the size of total stiffness matrix blocks. Each such vector is formed in sequence by use of the consistent load data, stored on disk from the first step of the execution (subroutining INPUT), and subsequently is stored on disk. Load vector elements

corresponding to unknown forces, namely reactions at the constraints, are replaced at this stage by zeros.

The boundary conditions imposed to the problem result in the presence of a number of unknown reactions, designated hereby F_i^* , and predefined displacements, designated by u_i^* . If such a boundary condition is applied to the i th element of the force and displacement vectors the system will have the form:

$$\begin{Bmatrix} F_1 \\ \vdots \\ F_i^* \\ \vdots \\ F_n \end{Bmatrix} = \begin{bmatrix} K_{11} & \dots & K_{1i} & \dots & K_{1n} \\ \vdots & & \vdots & & \vdots \\ K_{i1} & \dots & K_{ii} & \dots & K_{in} \\ \vdots & & \vdots & & \vdots \\ K_{n1} & \dots & K_{ni} & \dots & K_{nn} \end{bmatrix} \begin{Bmatrix} u_1 \\ \vdots \\ u_i^* \\ \vdots \\ u_n \end{Bmatrix}$$

This system is equivalent to a good degree of approximation to the system

$$\begin{Bmatrix} F_1 - K_{1i}u_i^* \\ \vdots \\ 0 \quad K_{ii}u_i^* \\ \vdots \\ F_n - K_{ni}u_i^* \end{Bmatrix} = \begin{bmatrix} K_{11} & \dots & K_{1i} & \dots & K_{1n} \\ \vdots & & \vdots & & \vdots \\ K_{i1} & \dots & B & \dots & K_{in} \\ \vdots & & \vdots & & \vdots \\ K_{n1} & \dots & K_{ni} & \dots & K_{nn} \end{bmatrix} \begin{Bmatrix} u_1 \\ \vdots \\ F_i^{**} \\ \vdots \\ u_n \end{Bmatrix}$$

Where B is a number of several orders of magnitude greater than the greater stiffness coefficient and $F_i^{**} = -\frac{F_i^*}{B}$

The products $F_i^{**}K_{ii}, \dots, F_i^{**}K_{ni}$ are negligible in size compared to the other coefficients. After the system is solved the unknown reaction F_i^* is determined by: $F_i^* = -BF_i^{**}$

This is a well known method of solution of mixed systems and it is felt that no further explanation is required [Ref. 5].

The process of application of the boundary conditions and transformation of the system to a form suitable for solution is performed by subroutine BCOND. In the case where all predefined displacements are equal to zero the process can be considerably simplified. In this case the final form of the system is:

$$\begin{Bmatrix} F_i \\ \vdots \\ 0 \\ \vdots \\ F_n \end{Bmatrix} = \begin{bmatrix} K_{11} & \dots & K_{1i} & \dots & K_{1n} \\ \vdots & & \vdots & & \vdots \\ \vdots & & \vdots & & \vdots \\ K_{i1} & \dots & B & \dots & K_{in} \\ \vdots & & \vdots & & \vdots \\ \vdots & & \vdots & & \vdots \\ K_{n1} & & K_{ni} & & K_{nn} \end{bmatrix} \begin{Bmatrix} u_1 \\ \vdots \\ F_i^{**} \\ \vdots \\ u_n \end{Bmatrix}$$

It is apparent that the only operation required is the substitution of the diagonal element K_{ii} by B .

For this reason subroutine BCOND was provided with two entries, one for the case of all zero predefined displacements and one (BCOND2) for the case of one or more nonzero predefined displacements. A number $B=10^{20}$ is used and is considered sufficient for the purpose of the program. The operations are carried out in a step by step procedure in a sense that at maximum two stiffness matrix coefficient blocks and their corresponding load vectors are simultaneously present in core.

6. Solution of the System - Determination of Reactions

The solution of the system is performed by subroutine SOLVE. The technique employed was devised by Professor Gilles Cantin of the Naval Postgraduate School and presented in Ref. 4.

In the process of solving the system of equations the inversion of diagonal blocks of the stiffness matrix is required. This inversion is performed by subroutine SYMINV. Before inversion of the block is performed it is checked whether it is singular or nearly singular. The trace of the matrix is evaluated (excluding diagonal elements containing the number $B=10^{20}$) and all diagonal elements are compared to the trace multiplied by a factor of 10^{-12} . If a diagonal element is found smaller than this product the block is characterized as nearly singular. The execution continues but a proper indication is given by SYMINV to SOLVE which in turn gives a warning message at the output via subroutine ERROR. The choice of the factor 10^{-12} was based on the number of significant digits provided by an IBM/360 machine on which the program was tested. Should the program be used on a different machine, alteration of this number should be considered in a sense that the difference between the exponent of the factor and the number of significant digits provided by the machine should be adequate to provide for loss of accuracy due to round off errors. An extensive presentation of the concepts underlying this idea is beyond the scope of this work.

If one diagonal element is found to be equal to zero the block is singular and further continuation of the execution is impossible. Again a proper indication is given by SYMINV to SOLVE which in turn gives an error message at the output via subroutine ERROR and the execution is stopped for this problem. Consequently the calling procedure is restarted from the beginning and the execution of the next problem begins (if such exists).

Upon inversion of each diagonal block $[A]$ of the stiffness matrix a condition number C is evaluated for this block within SYMINV. This condition number is defined as $C = \|A\| \cdot \|A^{-1}\|$ whereby $\|A\|$ is meant a norm of $[A]$. This condition number is printed out, and provides an upper bound on the expected relative error of the results of the problem. The difference between the number of significant digits provided by the machine used and the maximum exponent of the condition number gives approximately the number of significant digits in the displacement results. Further information on the significance of the condition number can be found in Ref. 6.

Following the solution of the system of equations the reactions are evaluated by subroutine DISP as explained in Section B5 but with an opposite sign. Consequently the nodal displacements and reactions are printed out by DISP.

7. Evaluation of Nodal Strains and Stresses

The nodal displacements as found by the solution of the system 2.5 are stored in vector form and in a sequence following the numbering convention of the nodes for the whole structure (Section II.A).

The determination of element nodal displacements by use of relation 2.2 requires a displacement vector containing the displacement components of all nodes of the corresponding element in a sequence following the numbering convention for individual elements (Section II.A). The formation of such vectors is accomplished by properly sorting the displacement components via subroutine DISK (entry WDISKB) called by DISP.

Subroutine STRESS evaluates the element nodal strains and stresses by use of relations 2.2 and 2.3. The evaluation of the product $[D]\{u\}$ requires the use of subroutine FORMK. The computational details of this operation are explained in Ref. 1.

Printing of element strains and stresses is optional since they provide useful information only for nodes on interfaces between elements of different materials.

Evaluation and printing of average nodal strains and stresses is performed by subroutine MAINE and is the concluding operation in the solution of each problem.

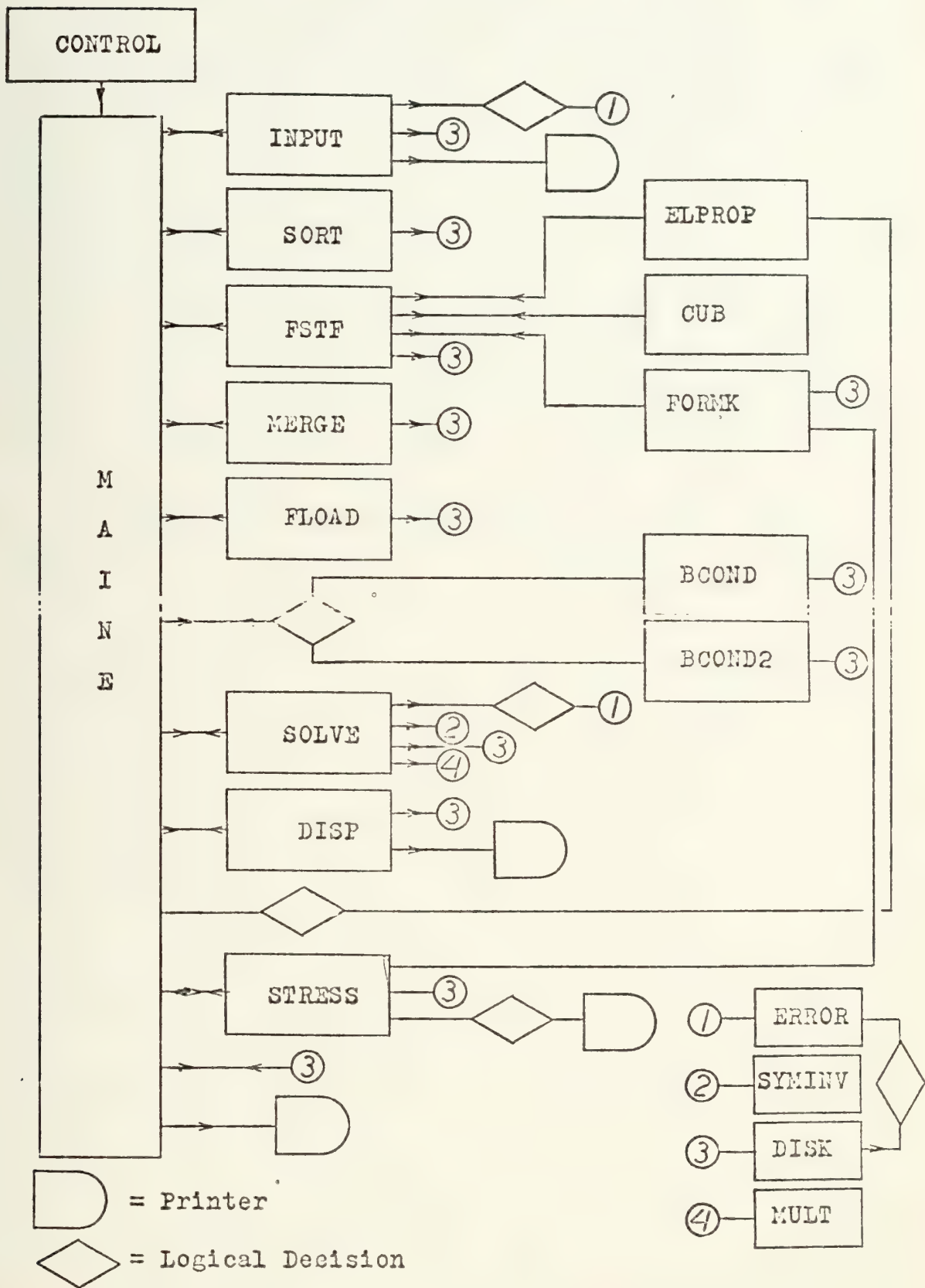
C. SUBROUTINE DISK

Transfer of data to and from the data files stored on direct access storage devices is accomplished by subroutine DISK. This subroutine uses multiple entries, each entry performing a different operation. Calling of DISK has been reduced to a minimum to avoid useless expense of execution time in data transfer. The most critical entries are those used for transferring element stiffness matrices and total stiffness matrix

blocks. The reason being the size of the arrays transferred and the number of transfers required. Reduction of the transfer time requires the use of the maximum possible record length. The size of element stiffness matrices is known for each type of element, thus the optimum record length is determined and the corresponding entries are coded on this basis. However, the block size in the equation solver is determined by the user and thus the optimum record length must be determined by the user (see Section IV). The function of the various entries is explained below. Entry names starting with the letter W denote transfer of data to a data file and names starting with the letter R denote transfer of data from data files to core.

RDDISK, WRDISK:	File number 7. Transfer of total stiffness matrix blocks.
RDISK1, WDISK1:	File number 8. Transfer of element stiffness matrices.
RDISKC, WDISKC:	File number 10. Transfer of nodal coordinate component vectors COORD (3)
RDSKR1, WDSKR1:	File number 20. Transfer of load vector blocks when stored in work vector RB1. Similarly for RDISKR2, RDISKR3, WDISKR2, WDISKR3.
RDISKB, WDISKB:	File number 21. Transfer of individual node displacement components vectors. The random access data file thus formed is used for the formation of element nodal displacement vectors.
RDISKS, WDISKS:	File number 12. Transfer of vector SSSS1 (See Appendix A.)

FUNCTIONAL FLOW DIAGRAM



IV. PROBLEM AND INPUT DATA PREPARATION

In this section the various steps in modeling a problem to be solved and the preparation of input data for the computer program are described.

A. STRUCTURE MODELING

The first step in applying a finite element solution to a problem is the selection of an appropriate mesh. In many cases an extrapolation of the results will be required and hence more than one meshes will have to be selected.

The mesh size selection does not follow any predetermined rules and will, in general, depend on the nature of the problem and the judgement of the analyst.

An arbitrary numbering convention must be adopted for the nodal points of the entire structure (in distinction with the numbering convention for the element nodes shown in Section II). The relation of node numbers in the two numbering systems establishes the connectivity matrix described in Section IV.B.1.

Though no restriction is imposed on the selection of the numbering convention for the entire structure, it has been shown in practice that an intelligent selection may considerably reduce the bandwidth of the total stiffness matrix.

B. INPUT DATA PREPARATION

Another factor to be determined by the analyst is the size of the blocks of coefficients in the equation solver. The block size affects the execution time required for solution of the system of equations by subroutine SOLVE. Some experimentation has been conducted on this subject by Cantin and is presented in Ref. 2. On the basis of this experimentation it seems that the larger the block size the lesser the execution time is. However, the subject has not, as yet, been thoroughly investigated and no firm rules can be stated at present. TRISOP allows the use of any block size. However, if a block size smaller than the size of the element stiffness matrix is selected, the dimensions of the various arrays of common block B3 may have to be changed.

After selection of the block size the number of records per block NREC7 and record length of file number 7 can be determined. Maximum transfer efficiency requires that NREC7 is minimum and thus the record length the maximum possible for the machine used. Subsequently the remaining input data can be determined. In the following a description of the non self-explanatory data is given.

1. Connectivity Matrix

The size of this matrix is $NEL \times NPEL$. Each row corresponds to one element and represents the correspondence of node numbers in the two numbering systems, namely, the element node numbering system and the structure node numbering

system. The values of the elements of each row are the node numbers in the entire structure system and are sequenced according to the element node numbering system. Thus, the second element of the fifth row represents node number two of finite element number five and its value is the number of this node in the structure system.

2. Nodal Point Coordinates

The nodal point coordinates are defined in the global system of reference. Cartesian or cylindrical coordinates may be used. These coordinate systems are defined in Fig.

3. All three coordinates of each node must be defined in the

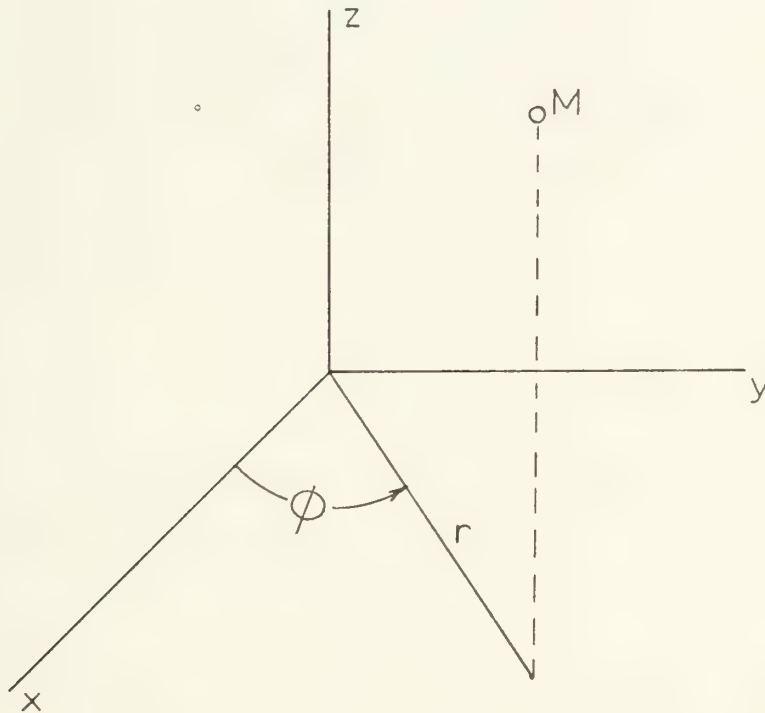


Figure 3.

same system. However, different nodes may be defined in different coordinate systems. Each set of nodal coordinates must be accompanied by a code number (KND) indicating whether cartesian or cylindrical coordinates are used.

3. Material Properties

The maximum permissible number of different materials is ten. In case the structure is composed of more than one materials a code number (NCON21) must be assigned to each material (NCON21=1,2,..., n, where n is the number of materials present in the structure). Material properties are given in the form of matrix ELDAT. Each row represents a different material and its elements are in sequence, Young's modulus, Poisson's ratio and coefficient of thermal expansion. The present form of TRISOP does not provide for thermal stress analysis and thus no value is required to be given to the coefficient of thermal expansion.

C. DATA CARDS

In order to preserve the flexibility of the input no reading formats are specified in the program except for titles, object time format arrays and general characteristics of the problem. Object time formats, stored in the array FMT, are used for reading input data. The same formats are used for echo-printing of the input data. In the description of input data cards the variable FMT denotes the object time format under which the corresponding data will be read. One data card is required per format. Formats may be arbitrarily

specified by the user. However it is recommended that only one data card be used for the variables described in Sections IV.C.5.c; 6.c; 7.c; 8.c; 9.c and 10.c. The number of cards required per group of data is based on the above recommendation.

Every group of data cards is preceded by a card containing alphanumeric information. Exception is made only for the connectivity matrix of cubic elements where two such cards are required. The contents of these cards are stored in the array TITLE and appear as identification headings at the output.

Both TITLE and FMT are read under a (10A8) format.

1. Characteristics of Data Files

These data are given only once for each run and are read under a (7I10) format. Six cards are required on which the values of record length and number of records for the various data files are given in the sequence: NR7, LRW7, NREC7, NR8, LRB8, LRW8, NR9, LRB9, NR10, LRB10, LRW10, NR12, LRB12, LRW12, NR13, LRB13, LRW13, NR14, LRB14, NR15, LRB15, NR16, LRB16, NR17, LRB17, NR19, LRB19, NR20, LRB20, LRW20, NR21, LRB21, LRW21, NR22, LRB22.

These cards can be produced by use of the auxiliary program JCL as described in Appendix B. The required values of the above variables are given in the same Appendix.

2. Initiation Card

Format: 1A8

Number of cards per problem: one

The word START must be punched starting at column one.

3. Problem Identification

Format: (10A8)

Number of cards per problem: one

Any alphameric information identifying the problem and punched in columns 2-80.

4. General Characteristics of the Problem

Format: (10I5)

Number of cards per problem: one

<u>Columns</u>	<u>Variable</u>	<u>Meaning</u>
1-5	NEL	Total number of elements
6-10	NDPT	Total number of nodal points
11-15	NS	Stiffness matrix block size
16-20	NPEL	Number of nodes per element
21-25	NMAT	Number of different materials
26-30	NCLD	Total number of nodes with concentrated loads
31-35	NPBC	Total number of nodes with boundary conditions
36-40	NPBC2	Number of nodes with non zero predefined displacements
41-45	NLINE	Number of lines to be printed per page for strain and stress tables
46-50	KEL	KEL=0: No printing of element nodal strains and stresses is required. KEL≠0: Printing of above is is required.

5. Connectivity Matrix

a. TITLE

b. FMT

c. Connectivity data.

Number of cards: NEL

Variables: NCON= Array of connectivities for one
element (one row of connectivity
matrix)

NCON21= Material code number for the
corresponding element (if required)

The connectivity matrix is read row-wise one row at a
time. In the case of structures composed of more than one
material, the last element of NCON must be followed by NCON21.

6. Nodal Coordinates

a. TITLE

b. FMT

c. Coordinate data

Number of cards: NDPT

Variables: IEL= Node number in the entire struc-
ture numbering system.

COORD= Array of coordinates of node IEL
in the global system. Coordinates
are given in the sequence x, y, z
or r, ϕ , z.

KND= Code number indicating cartesian
or cylindrical coordinates.

KND=0: Cartesian coordinates

KND \neq 0: Cylindrical coordinates.

7. Material Properties

a. TITLE

b. FMT

c. Material property data

Number of cards: NMAT (maximum ten)

Variables: N= Material code number

ELDAT= Array of material properties.

8. Nodal Consistent Load Data

a. TITLE

b. FMT

c. Load data

Number of cards: NCL= Node number in the numbering
system of the entire struc-
ture

CLOAD1= Array of concentrated load
components in the global
system. Load components
are given in the sequence
x,y,z.

Load data cards must be in ascending order of NCL
values.

9. Boundary Condition Codes

a. TITLE

b. FMT

c. Codes

Number of cards: NPBC

Variable: NBC= Array of boundary condition codes
for each node. The first element
is the node number and the remain-
ing three represent the boundary

conditions applied in the x,y,z
direction respectively.

The codes employed are:

0: Free node.

1: Node constrained to zero
displacement.

2: A predefined displacement
exists.

The boundary condition code data cards must be
sequenced in ascending order of node number.

10. Predefined Displacements (if required)

a. TITLE

b. FMT

c. Displacement data.

Number of cards: NPBC2

Variables: NDBC = Node number.

BCON = Array of predefined displacement
components in the global system,
given in the x,y,z sequence.

Predefined displacement data cards must be sequenced in
ascending order of the values of NDBC.

In case no predefined displacements exist in a problem
(NPBC2=0) all cards described in this paragraph must be omitted.

11. Subsequent Problems - Stop Trap for Last Problem

In case more than one problem is to be executed in
one run, the data information described in paragraphs IV.C,2
to IV.C.10 must be provided for each problem. The initiation

card (START) of each problem must immediately follow the last card of the previous problem.

The last problem must be followed by the following cards which serve as a stop trap.

- a. One card with the word "START" starting in column one.
- b. One card containing any alphanumeric information (or blank).
- c. One card with any negative integer punched in columns 1-5.

In order to provide for a better understanding of the above documentation sample inputs for two problems, one utilizing quadratic and one cubic elements are presented in Appendix C.

D. DIRECT ACCESS DATA FILES REQUIREMENTS

The method of specification for the direct access file requirements will vary from one installation to the other. The particular features of the different installations will affect the form of subroutine DISK and the JCL cards. In Appendix B these requirements are described for the IBM 360/67 operating system. The space requirements for each file as described in Appendix C are those to be used in establishing the form of subroutine DISK and JCL cards for any machine.

V. SOLUTION OF A SIMPLE PROBLEM

When the initial form of TRISOP, for quadratic elements, was coded by Professor Cantin, it was tested on simple problems, and results were obtained and compared with classical solutions.

Testing of the present development of TRISOP was based on these problems and no further experimentation was conducted.

In order to show the function of the program, as well as some observations affecting the use of it, the solution of a simple problem is presented in this section.

The problem selected is that usually presented in classical texts of strength of materials as a simply supported beam of uniform cross section subjected to a concentrated load in the middle of the span.

The model used for this problem for the quadratic element formulation is shown in Fig. 4. The concentrated load is substituted by a line load. Nodes 1, 2 and 3 (here arbitrarily numbered) are constrained to zero displacement in all directions and nodes 4, 5 and 6 are allowed to move in the z direction only. The model for cubic elements is the same with the difference that the constrained nodes are four at each end instead of three.

The consistent loads corresponding to the line load applied were found as required by the program. Such loads appear only at the nodes of the midsection of the beam and are shown in Fig. 6.

Five different meshes were employed in order to show the variation of results with mesh size. The meshes chosen were 2, 4, 6, 8 and 10 elements in the z direction for both quadratic and cubic elements.

Numerical integration was carried by use of 2, 3 and 4 Gauss points (in different runs) for quadratic elements and 3, 4 and 5 points for cubic elements. Listings of the subroutines used are shown in Appendix I.

Integration with 3 and 4 Gauss points for quadratic elements and 4 and 5 points for cubic elements gave essentially the same results for this problem. For problems using elements with curved boundaries the results will vary with the number of Gauss points used. The solution with quadratic elements using 2 integration points was numerically unstable and no results were obtained. Numerical results for the maximum displacement at the midsection are given in Tables I and II. The same results are shown graphically in Fig. 7 for both quadratic and cubic elements. In the same figure, the execution time for subroutine FSTF and SOLVE, which consume most of the execution time for each problem, is shown. Figure 8 shows graphically the displacements and execution time obtained from the solution using cubic elements.

A. ALTERNATE FORMULATION

In order to observe some further features of the finite element solution, the same problem was solved formulated in a different way.

By use of the existing symmetry in the z direction, only half of the beam was considered as shown in Fig. 5. All nodes lying in the plane of symmetry were constrained to a zero displacement in the x direction. The presence of additional constraints enabled the use of two integration points. Numerical values for the maximum displacement in the midsection are shown in Table III. The same results as well as execution time are shown in graphical form in Fig. 9.

B. DISCUSSION OF RESULTS

Observation of the results presented here reveals that mesh refinement leads to improved results which eventually will converge to a certain value. Use of less accurate numerical integration by reducing the number of integration points leads to improved results and considerable reduction of execution time. Also the use of existing symmetry and in general different formulations of the problem may prove to be beneficial.

Results obtained by use of cubic elements are considerably better than those of quadratic elements. However the execution time is appreciably higher for cubic elements and it is a matter of judgement whether cubic or quadratic elements should be used for each specific problem.

These are observations essentially made on the results of a single and basically simple problem. Thus no firm rules regarding the use of TRISOP can be established and further experimentation with different problems has to be conducted for this purpose.

TABLE I

Quadratic Elements - Maximum Displacements in $\times 10^{-4}$

ELEM.	2	4	6	8	10
CUB 3 or CUB 4	5.761	6.847	6.992	7.026	7.038

TABLE II

Cubic Elements - Maximum Displacements in $\times 10^{-4}$

ELEM.	2	4	6	8	10
CUB 5 or CUB 4	7.045	7.055	7.062	7.067	7.070
CUB 3	7.045	7.067	7.068	7.069	7.072

TABLE III

Quadratic Elements - Alternate Formulation

Max. Disp. in $\times 10^{-4}$

ELEM.	2	4	6
CUB 4 or CUB 3	5.761	6.847	6.992
CUB 2	7.191	7.066	7.053

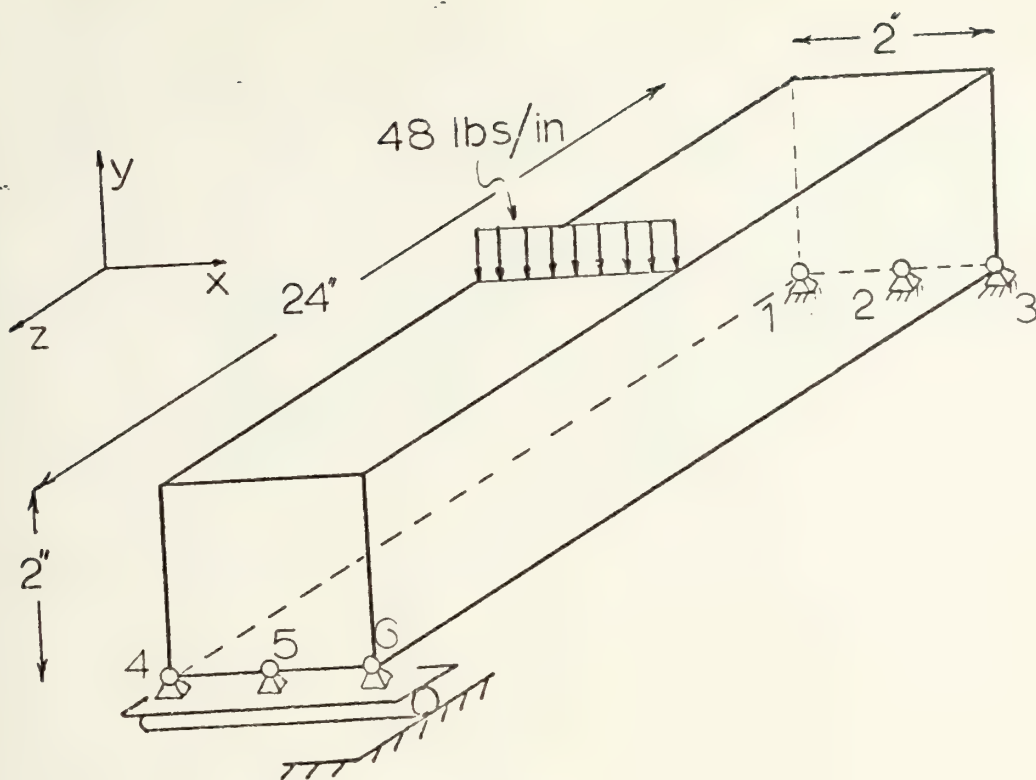


Figure 4

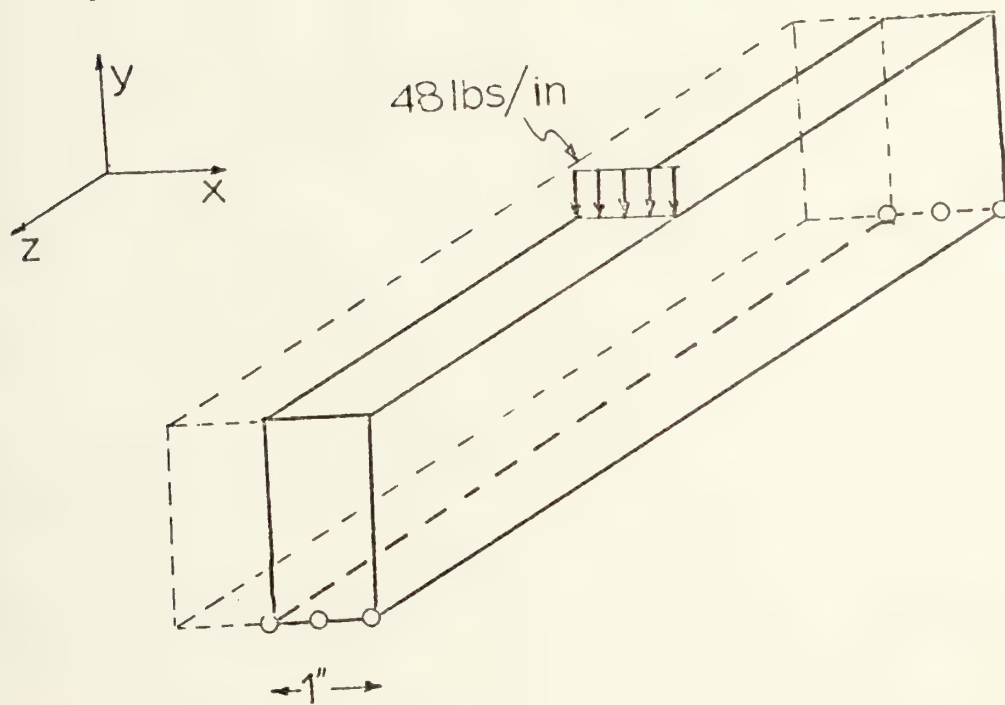
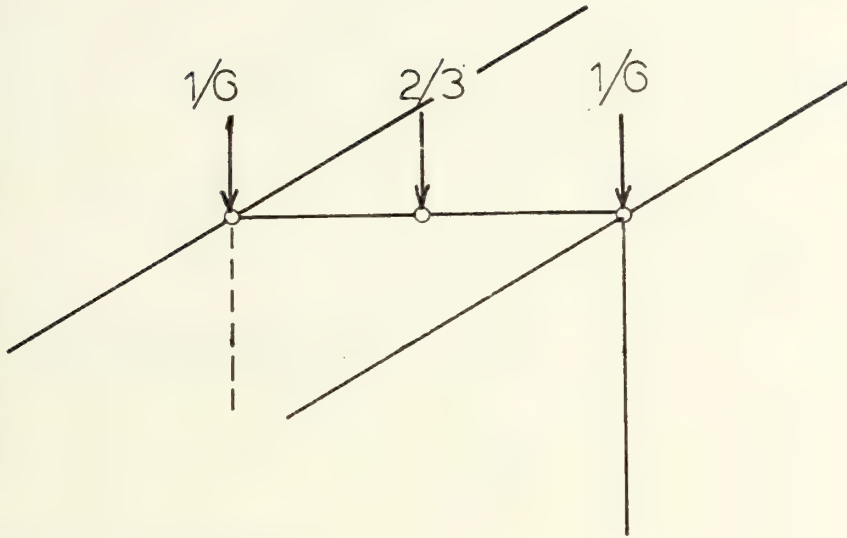
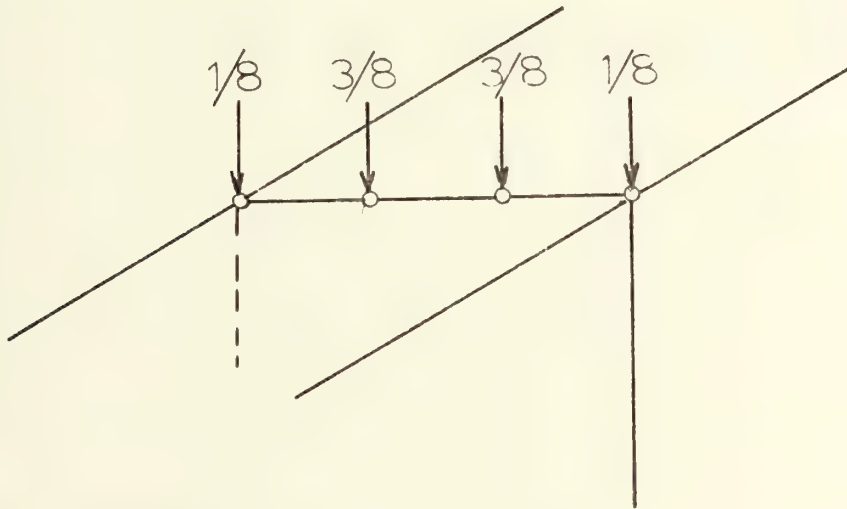


Figure 5



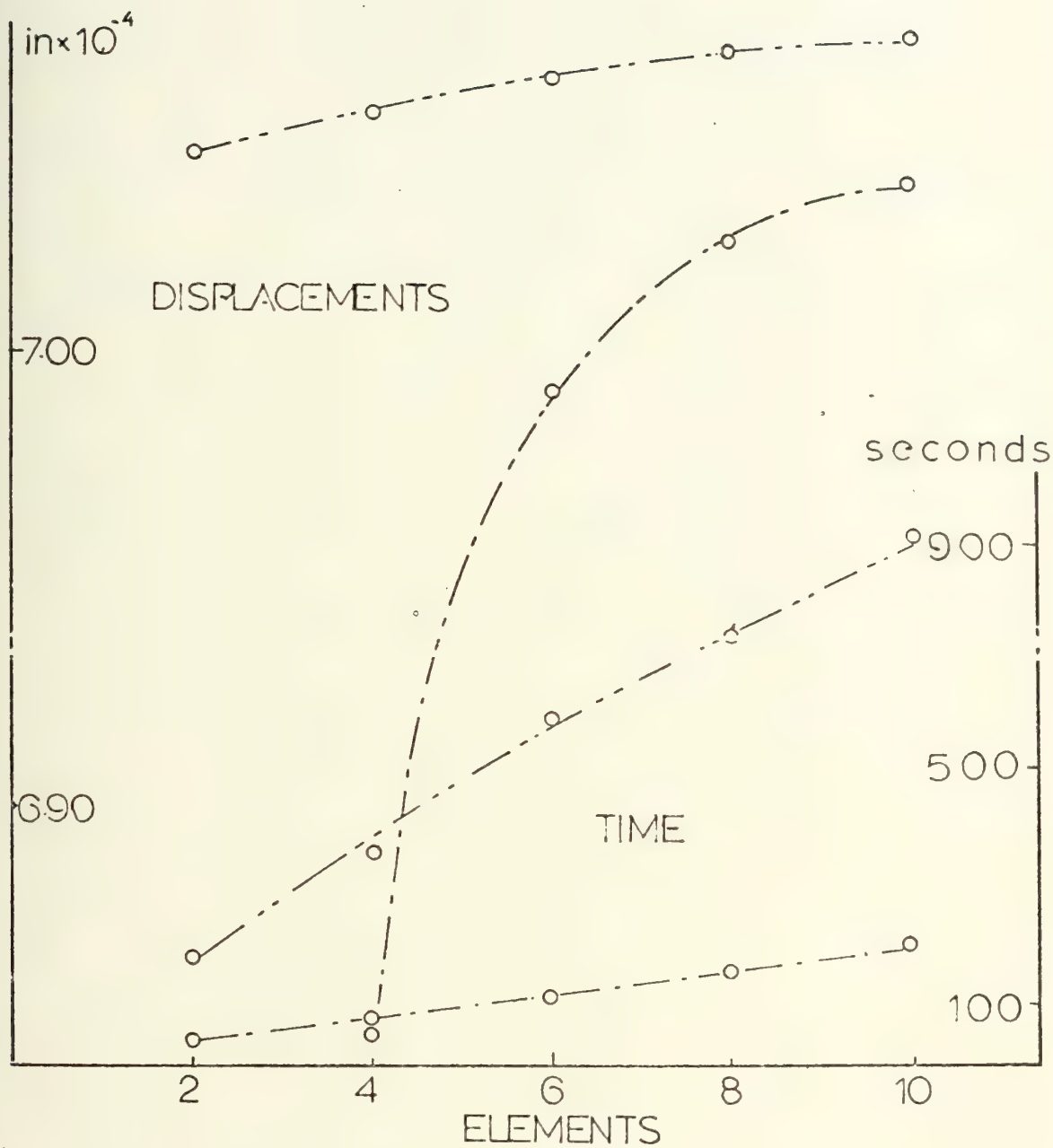
Quadratic elements



Cubic elements

Numbers represent proportions of total load applied on each node.

Figure 6



- - - - Cubic el. - CUB 4
 - . - - Quadr. el. - CUB 3

Figure 7

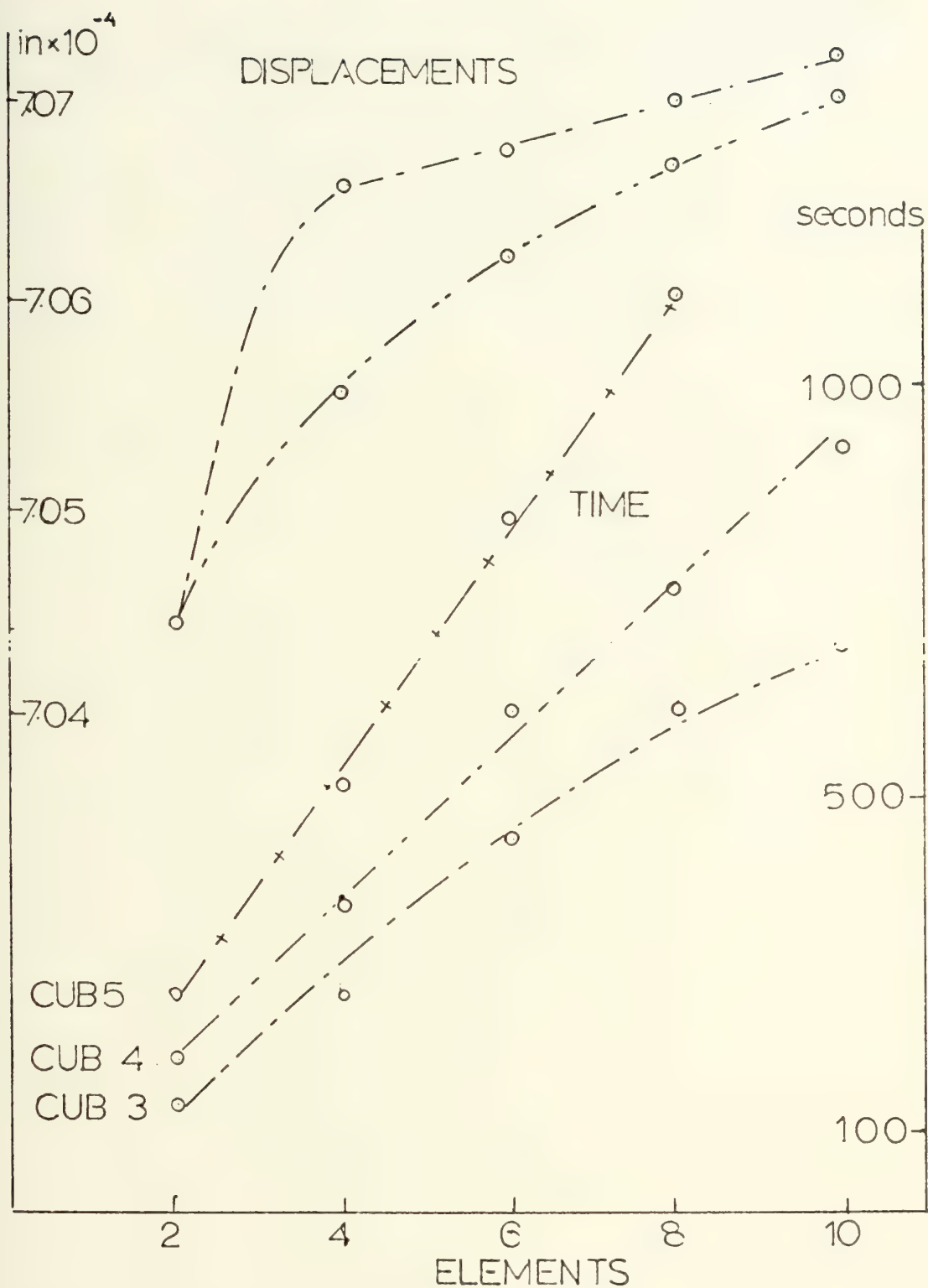


Figure 8

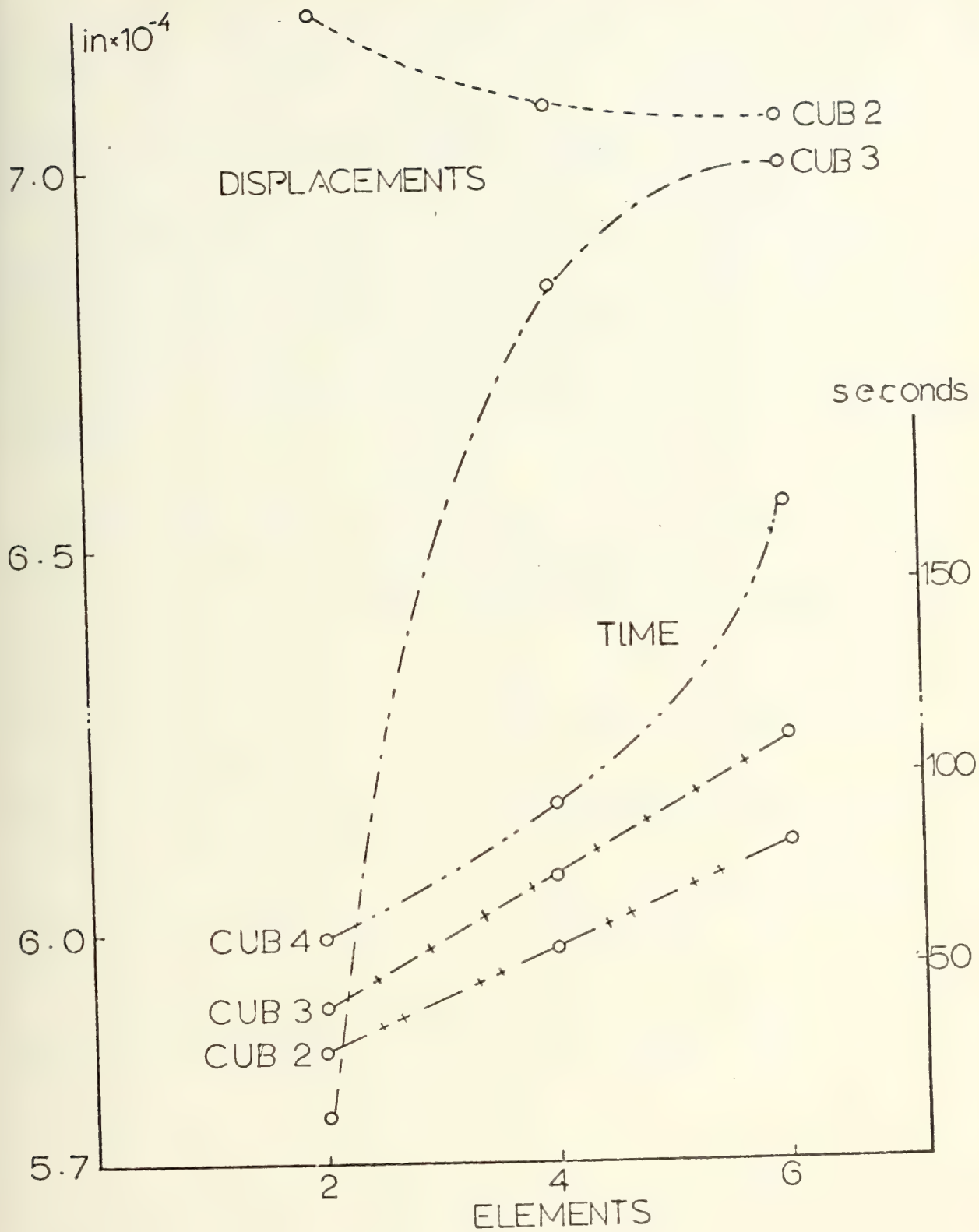


Figure 9

APPENDIX A

COMPUTER PROGRAM VARIABLES

In this Appendix the most important variables appearing in the programs are listed together with a brief description of their function. The variables of each group are sequenced in alphabetic order. Arrays are shown with the dimensions required.

A. DATA FILES CHARACTERISTICS

NRX : Number of records of file number "X"

LRBX : Length of records of file number "X" measured in bytes

LRWX: : Length of records of file number "X" measured in single words.

NREC7 : Number of records per total stiffness matrix block

NRX, LRBX AND LRWX are used only for checking the adequacy of disk storage space requested by the JCL cards.

NREC7 specified by the user, is used in subroutine DISK and controls the transfer of stiffness matrix blocks.

B. VARIABLES APPEARING IN 'COMMON' STATEMENTS

AK1(NS,NS) : Work space used primarily for storage of stiffness matrix blocks when transferred in core. Alternatively is used for storing

equivalenced arrays and thus reduce the space requirements. Similarly for AK2 and AK3.

- BCON(3) : Array of non zero predefined displacement components for one node. It is used only in entry BCOND2 of subroutine BCOND. The complete array of non zero predefined displacements is stored on disk and only the components for one node at a time are present in core.
- CLOAD(4) : Concentrated load data for one node. The first three elements represent the components of the load and the fourth the node number. It is used only in subroutine FLOAD. The complete array of concentrated load data is stored on disk and only data for one node at a time are present in core.
- COORD(3) : Array of global coordinates for one node. The complete array of nodal coordinates is stored on disk in node sequence following the numbering convention for the whole structure. Coordinates of only one node at a time are present in core.
- COREL(NPEL,3) : Array containing the global coordinates for all nodes of one element. These arrays are formed by SUBROUTINE SORT one at a time and subsequently stored on disk. Only one such array is present in core every time.

ELAST(616) : Matrix of elastic constants of elements formed by subroutine ELPROP.

ELDAT(10,3) : Array containing information on properties of materials composing the structure. It is used in ELPROP for the formation of ELAST.

KEL : Index number indicating whether printing of element nodal strains and stresses is required or not.

MM : Number of stiffness matrix blocks per row (see Ref. 2)

NBC(4) : Boundary condition information for one node. The first element represents the node number and the remaining the boundary condition codes. The complete set of such arrays is stored on disk in node number sequence and only one is present in core at a time.

NBCH : Highest node number at which a boundary condition is applied.

NBCL : Lowest node number at which a boundary condition is applied.

NCLD : Total number of nodes at which a concentrated load is applied.

NCON(NPEL) : Array of connectivities for one element. This array shows the correspondence of element node numbers in the two numbering systems employed, namely, element numbering system and entire structure numbering system. The values of

the elements of the array represent the node numbers in the structure numbering system and are sequenced according to the element numbering system. Thus the third element of the array represents node number three of the element and its value represents the number of this node in the structure numbering system. The complete set of these arrays is stored on disk and only one at a time is brought into core.

- NCON21 : Indicates the material of the corresponding element and is used in the case the structure is composed of more than one material. The values of this variable must conform with the material number given in array ELDAT.
- ncount : Number of coefficients per stiffness matrix block.
- NDF : Number of spacial degrees of freedom. In the present form of the program its value is constant and equal to three. However, it is used throughout the origin as a variable in order to provide for future developments of the programs as for example degeneration to two dimensional elements.
- NDFP : Constant equal to: $NDF+1$
- NDPT : Total number of nodal points in the structure.
- NEL : Total number of finite elements used in the problem.

NEQ : Total number of equations of system (2.5).

NLINE : Number of lines to be printed per page in the output for displacement, reactions, strains and stresses.

NMAT : Number of materials of which the structure is composed.

NN : Number of stiffness matrix blocks per column (See Ref. 2).

NPBC : Total number of nodes at which boundary conditions are applied.

NPBC2 : Total number of nodes at which non zero predefined displacements are applied.

NPEL : Number of nodes per element.

NREC7 : Number of records per stiffness matrix block.

NS : Stiffness matrix block size.

NST : Size of element stiffness matrix.

NSTF : Number of coefficients per element stiffness matrix.

SSSS1(7) : Work vector used for transferring to and from data files the algebraic sums of stress or strain components contributed by elements sharing a common node. The seventh element of the array is used to count the number of occurrences of the node. Such arrays are formed in subroutine STRESS and further used in MAINE for the evaluation of average strains and stresses.

UJNT(3) : Array of displacement components for one node. The complete set of such arrays is stored on disk and

displacements for only one node at a time
are present in core.

C. OTHER IMPORTANT VARIABLES

In order to avoid redundant definitions, the most important variables of each subroutine that have not been presented in Sections A and B of this Appendix are explained below.

1. Subroutine ELPROP

E : Young's modulus
PR : Poisson's ratio

2. Subroutine CUB

STK(NST,NST) : Element stiffness matrix
X,Y,Z : Gauss points coordinates

3. Subroutine FORMK

AJ(NDF,NDF) : Jacobian matrix
AK(NST,NST) : Integrant of stiffness matrix integral
formula
B(6,NST) : Matrix $[N^*] = [D][N]$
CORDG(NPEL,NDF) : Element nodal coordinate array in the local
system
DNX(NDF,NPEL) : Array of derivatives of shape functions with
respect to x,y,z .
W1(NDF,NPEL) : Array of derivatives of shape functions with
respect to ξ,η,ζ .

4. Subroutine BCOND

NBC : Stiffness matrix row number corresponding to
the 'z' component of the boundary condition
examined.

IBL,IBH : Lowest and highest row number in the block examined.

INDEX : Number of boundary conditions applied.

AKW1(NCOUNT)
AK1(NS,NS) : Diagonal stiffness matrix block.

AKW2(NCOUNT)
AK2(NS,NS) : Offdiagonal stiffness matrix block.

RB1(NS) : Load vector corresponding to row 'I'.

RB2(NS) : Load vector corresponding to blocks symmetric to the offdiagonal blocks of row 'I'.

MM1 : Specifies the extreme offdiagonal block to be examined in each row in order to avoid blank blocks.

IFR : Specifies whether any boundary conditions have to be applied to the diagonal block of row 'I' or not.

5. Subroutine SOLVE

KMM : Same as MM1 in MERGE.

IFLG : Indicates whether block AK2 is singular or nearly singular.

6. Subroutine DISP

JLL : Indicates the number of non blank elements in the last load vector

7. Subroutine STRESS

CORDG(NPEL,3) : Vector of element nodal coordinates in the local system.

SNELM(NPEL,6) : Array of element nodal strains.

SSELM(NPEL,6) : Array of element nodal stresses.

SSNN(6) : Strain vector for one node.
SSSS(6) : Stress vector for one node.
UEL(NST) : Vector of element nodal displacements.

APPENDIX B

DIRECT ACCESS DATA FILES REQUIREMENTS

In this Appendix the direct access data file requirements as well as the necessary control cards for use of the compiler of release 18 of an IBM 360 operating system are described.

A. DATA FILE SPACE REQUIREMENTS

The data file space requirements are described by the variables NRX, LRBX, LRWX as they are explained in Section A of Appendix A.

The required values of these variables are as follows:
(the values of LRWX are not presented since always $LRWX = LRBX/4$)

NR7	=	MM x NN x NREC7
LRB7	=	NCOUNT x 8/NREC7
NR8	=	NEL x 4 for quadratic elements NEL x 12 for cubic elements
LRB8	=	7200 for quadratic elements 6144 for cubic elements
NR9	=	NEL
LRB9	=	80 for quadratic elements 128 for cubic elements
NR10	=	NDPT
LRB10	=	24

NR12 = NDPT

LRB12 = 48

NR13 = NDPT

LRB13 = 48

NR14 = NCLD

LRB14 = 32

NR15 = NPBC

LRB15 = 16

NR16 = NEL

LRB16 = 4

NR17 = NPBC

LRB17 = 32

NR19 = NEL

LRB19 = 480

NR20 = NN

LRB20 = NS x 8

NR21 = NN x NS/NDF

LRB21 = 24

NR22 = NPBC2

LRB22 = 24

B. CONTROL CARDS

Job control language cards for an IBM 360 operating system are of the form:

```
GO.FTXXF001 DD UNIT = SYSDA, SPACE = (LRBX, (NRX,1)),DISP=
      (NEW,DELETE)
```

where X and XX represent the number of the corresponding file.

These cards may be produced by the auxiliary computer program named JCL and whose listing is presented in Appendix H. This program is capable of printing and punching the JCL cards as well as the data files characteristics (Section IV.C.1) for a series of problems in one run. One data card is required for each problem. This card must contain the variables: NEL, NPEL, MM, NS, NDPT, NPBC, NPBC2, NCLD, NREC7, NMAT punched in format (1X,1015). Following the data card of the last problem, a card with a negative integer punched in columns one to six right justified, must be given. The control cards required for this program in order to be used in an IBM 360 OS are shown in Appendix F.

In case the value of MM is not known, it can be computed by the program from connectivity data. In this case MM is given the value zero in the data card described above and which must be immediately followed by the data cards:

1. Object Time Format (20A4)

Number of cards: one

The format under which connectivity data will be read is punched. This format must provide for reading NPEL integers at a time, representing one row of the connectivity matrix.

2. Connectivity Data

Number of cards: NEL x (number of cards per row of connectivity matrix as per specified format.)

Each individual row of the connectivity matrix is punched in sequence under the specified format.

The input data of program JCL are printed in the output for checking purposes. A sample output is presented in Appendix E.

C. DEFINE FILE STATEMENTS

The DEFINE FILE statements contained in subroutine DISK and defining the space requirements for random access files must be adjusted to the needs of the larger problem of each run. The form of these statements is:

```
DEFINE FILE X(NRX,LRWX,U,IX)
```

where X represents the number of the corresponding file.

These statements are also produced by program JCL as described in Section B of this Appendix.

SAMPLE INPUT DATA DECKS FOR TRISOP

[illegible]

B. CUBIC ELEMENTS

72	6144	1536	12	24	6144	1536	KND
2	128	52	24	6	52	56	
16	52	56	14	4	32	8	
0	0	0	32	24	2	480	
0	768	192	64		6	0	
START SIMPLY SUPPORTED BEAM - CUBIC ELEMENTS - MESH 1X1X2							
1	2	1	8	60	1	0	CONN. MTR.
96	32	4	8	M	N	P	
A	E	H	J	L	D	E	
B	V	O	O	C	I	F	
C	U		K	A			
S			A	B			
7	3	4	11	10	15	13	Z
5	2	6	12	24	28	32	
18	29	21	23	31	35	33	
25	22	26	32	44	48	52	
38	47	41	43	46	50	54	
JOINT	X			Y			
COORD.							
(10X,16,3F18.5,15)							
1	0.0	0.0	2.0	0.0	24.0	0.0	
2	0.0	0.0	1.3	3333	24.0	0.0	
3	0.0	0.0	0.6	6667	24.0	0.0	
4	0.0	0.6	0.0	0000	24.0	0.0	
5	0.0	6667	0.0	0000	24.0	0.0	
6	0.6	6667	0.0	0000	24.0	0.0	
7	1.3	3333	2.0	0000	24.0	0.0	
8	1.3	3333	0.0	0000	24.0	0.0	
9	2.0	0000	1.3	3333	24.0	0.0	
10	2.0	0000	0.6	6667	24.0	0.0	
11	2.0	0000	0.0	0000	24.0	0.0	
12	0.0	0.0	2.0	0000	20.0	0.0	
13	0.0	0.0	0.0	0000	20.0	0.0	
14	0.0	0.0	0.0	0000	20.0	0.0	
15	0.0	0.0	0.0	0000	16.0	0.0	
16	0.0	0.0	0.0	0000	16.0	0.0	
17	0.0	0.0	0.0	0000	16.0	0.0	
18	0.0	0.0	0.0	0000	12.0	0.0	
19	0.0	0.0	0.0	0000	12.0	0.0	
20	0.0	0.0	0.0	0000	12.0	0.0	
21	0.0	0.0	0.0	0000	12.0	0.0	
22	0.0	0.0	0.0	0000	12.0	0.0	
23	0.0	0.0	0.0	0000	12.0	0.0	
24	0.0	0.0	0.0	0000	12.0	0.0	
25	0.0	0.0	0.0	0000	12.0	0.0	
26	0.0	0.0	0.0	0000	12.0	0.0	
27	0.0	0.0	0.0	0000	12.0	0.0	
28	0.0	0.0	0.0	0000	12.0	0.0	
29	0.0	0.0	0.0	0000	12.0	0.0	
30	0.0	0.0	0.0	0000	12.0	0.0	

	Y.MOD.	P.RATIO	ALPHA	YLD	ZLD	CONCENTRATED LOAD DATA	ELEMENT DATA
31	1X,1I3,1X,1F10.0,2F10.5)		2.00000			0.66667	12.00000
32	1 30000000.00.3		2.00000			0.00000	12.00000
33	N.PT		0.00000			0.00000	8.00000
34	(5X,1I5,3F10.5)		2.00000			0.00000	8.00000
35			0.00000			0.00000	8.00000
36			0.00000			0.00000	4.00000
37			0.00000			0.00000	4.00000
38			2.00000			0.00000	4.00000
39			0.00000			0.00000	0.00000
40			0.00000			1.33333	0.00000
41			0.00000			0.66667	0.00000
42			0.00000			0.00000	0.00000
43			0.66667			0.00000	0.00000
44			0.66667			0.00000	0.00000
45			1.33333			0.00000	0.00000
46			1.33333			0.00000	0.00000
47			2.00000			0.00000	0.00000
48			2.00000			1.33333	0.00000
49			2.00000			0.66667	0.00000
50			2.00000			0.00000	0.00000
51			2.00000			0.00000	0.00000
52			2.00000			0.00000	0.00000

	X	Y	Z	BOUNDARY CONDITIONS
(4I10)	1 1 1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1 1 1		
			1 1 1 1 1 1 1 1 1 1	

START

-1

APPENDIX D

SAMPLE OUTPUT FOR TRISOP

A sample output for the solution of a problem by use of quadratic elements is presented in this Appendix. The problem chosen is that of a simply supported beam with a line load acting on the mid-section and discretized in a 1x1x2 mesh. This problem is one of those discussed in Section V.

The form of the output for a solution using cubic elements will be identical except for the connectivity matrix. The form of the connectivity matrix for the same problem solved by use of cubic elements will be as follows:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
Q	R	S	T	U	V	W	Y	Z	Al	Bl	Cl	Dl	El	Fl	G1
9	7	5	1	2	3	4	6	8	12	11	10	15	13	14	16
19	17	18	20	29	27	25	21	22	23	24	26	28	32	31	30
29	27	25	21	22	23	24	26	28	32	31	30	35	33	34	36
39	37	38	40	49	47	45	41	42	43	44	46	48	52	51	50

SIMPLY SUPPORTED BEAM - QUADRATIC ELEMENTS - MESH 1X1X2

NEL...TOTAL NUMBER OF ELEMENTS..... 2
NDPT...TOTAL NUMBER OF NODAL POINTS..... 32
NMAT...NUMBER OF DIFFERENT MATERIALS..... 1
 (MAXIMUM IS 10)
NS....BLOCK SIZE FOR THE LARGE CAPACITY SOLVER..... 60
NPRC...NUMBER OF NODAL POINTS WITH ROUNDED COND..... 6
NPRC?...NUMBER OF NODAL POINTS WITH PREDEF. DISPL..... 0
NCLO...NUMBER OF NODAL POINTS WITH CONCENT. LOAD..... 3

A B C D E F G H I J K L M N O P Q R S T CONN. MTR.

6 4 1 2 3 5 8 7 11 9 10 12 18 16 13 14 15 17 20 19
18 16 13 14 15 17 20 19 23 21 22 24 30 28 25 26 27 29 22 31

NEQ...TOTAL NUMBER OF EQUATIONS..... 96
NBRAND...HALF-BAND WIDTH OF THE SYSTEM..... 60
NN....NUMBER OF BLOCKS PER COLUMN..... 2
NR....NUMBER OF BLOCKS PER ROW..... 2
NCOUNT...NUMBER OF COEFFICIENTS PER BLOCK..... 3600
NSTF...NUMBER OF COEFFICIENTS PER ELEMENT..... 3600

COORD.	JOINT	X	Y	Z	KND
	1	0.0	2.00000	24.00000	0
	2	0.0	1.00000	24.00000	0
	3	0.0	0.0	24.00000	0
	4	1.00000	2.00000	24.00000	0
	5	1.00000	0.0	24.00000	0
	6	2.00000	2.00000	24.00000	0
	7	2.00000	1.00000	24.00000	0
	8	2.00000	0.0	24.00000	0
	9	0.0	2.00000	18.00000	0
	10	0.0	0.0	18.00000	0
	11	2.00000	2.00000	18.00000	0
	12	2.00000	0.0	18.00000	0
	13	0.0	2.00000	12.00000	0
	14	0.0	1.00000	12.00000	0
	15	0.0	0.0	12.00000	0
	16	1.00000	2.00000	12.00000	0
	17	1.00000	0.0	12.00000	0
	18	2.00000	2.00000	12.00000	0
	19	2.00000	1.00000	12.00000	0
	20	2.00000	0.0	12.00000	0
	21	0.0	2.00000	6.00000	0
	22	0.0	0.0	6.00000	0
	23	2.00000	2.00000	6.00000	0
	24	2.00000	0.0	6.00000	0
	25	0.0	2.00000	0.0	0
	26	0.0	1.00000	0.0	0
	27	0.0	0.0	0.0	0
	28	1.00000	2.00000	0.0	0
	29	1.00000	0.0	0.0	0
	30	2.00000	2.00000	0.0	0
	31	2.00000	1.00000	0.0	0
	32	2.00000	0.0	0.0	0

NMAT	Y.MOD.	P.RATIO	ALPHA	ELEMENT DATA
1	30000000.	0.30000	0.0	



N.PT XLD YLD ZLD CONCENTRATED LOAD DATA

13 0.0 -16.00000 0.0 0.0

16 0.0 -64.00000 0.0 0.0

18 0.0 -16.00000 0.0 0.0

J1 X Y Z BOUNDARY CONDITIONS

3 1 1 0

5 1 1 0

9 1 1 0

27 1 1 1

28 1 1 1

32 1 1 1

***** INPUT *****

TIME = 14.62 SECONDS(CPU = 0.40 SECONDS)

***** SPLIT *****

TIME = 5.86 SECONDS(CPU = 0.26 SECONDS)

***** FSTF *****

TIME = 27.59 SECONDS(CPU = 15.61 SECONDS)

***** MERGE *****

TIME = 17.20 SECONDS(CPU = 3.21 SECONDS)

***** FLOAD *****

TIME = 5.50 SECONDS(CPU = 0.54 SECONDS)

***** PCOND *****

TIME = 4.63 SECONDS(CPU = 1.20 SECONDS)

CONDITION NUMBER 2.4258750813639340D 04 2.8080341880341820D 09 8.635051090265440D-06

CONDITION NUMBER 2.437251322157370D 04 1.0012330579315250D 09 2.4339946033916030D-05

***** SOLVE *****

TIME = 39.82 SECONDS(CPU = 27.11 SECONDS)

D-I-S-P-L-A-C-F-M-E-N-T-S

N.P.T

X

Y

Z

-1.61088015031574100-06	-1.550900624602768700-06	3.44636096172503600-07
-2.27128617666770400-07	-1.7404130631518200-06	8.9210668817684000-05
0.0	0.0	1.72165169122805000-04
-5.19711067538425600-18	-6.0339555091583700-07	2.71719568077456800-07
0.0	0.0	1.72335510347767100-04
1.61088015030614100-06	-1.550900624602769000-06	3.44636096144788500-07
2.27128617662963600-07	-1.74041306314681300-06	8.9210668817408700-05
0.0	0.0	1.72165169122775400-04
-2.20441064927391000-06	-4.17684492023517700-04	3.75608945729575300-05
2.17340379716154100-06	-4.17309297436146700-04	1.35013780892295700-04
2.20441064910861700-06	-4.172684492023485100-04	3.75608945729458600-05
-2.17340379725621000-06	-4.17309297436117000-04	1.35013780892283700-04
-2.174416749064152200-06	-5.76053690506871500-04	8.614256085980453800-05
-3.14305456129398400-07	-5.77050637049747700-04	8.6143592969701100-05
-2.16995397579002400-06	-5.75584438974499500-04	8.614313100717902800-05
-1.15560016046181900-16	-5.74362056754158900-04	8.61436190604985800-05
-2.171800339854986000-17	-5.73812476086323700-04	8.611093034362500-05
2.14305455986186900-06	-5.76053690506773700-04	8.61256085980525300-05
-2.169695337584419700-06	-5.77050637049747700-04	8.6143592969778500-05
-2.20483558823372000-06	-4.17356709891503500-04	8.61613100717981200-05
2.16645000916697600-06	-4.17356709891503500-04	1.34724787642083400-04
2.20483558811726000-06	-4.17356709891503500-04	1.34724787642083400-04
-2.16945000919500300-06	-4.17356709891503500-04	1.34724787642083400-04
-1.61345271328767100-06	-1.54993567041465300-06	1.72312437429853700-05
-2.223675256023195300-07	-1.74028516702261400-06	1.71974929154010100-04
0.0	0.0	8.603136722668555600-05
-2.25798951501906600-18	-6.03514768682409200-07	0.0
0.0	0.0	1.71975188704856400-04
1.61345271328326600-06	-1.54993567040935500-06	0.0
2.223675256021414800-07	-1.74028516701911100-06	1.71974929554022600-04
0.0	0.0	8.603136722668637300-05

R-E-A-C-T-I-O-N-S

N.PT	X	Y	Z
3	2.3658853372884870D 01	-1.5091973755734090D 01	0.0
5	2.6762408498878810D-11	-1.7816052489112260D 01	0.0
8	-2.3658853372945610D 01	-1.5091973755648660D 01	0.0
27	2.4127265073037160D 01	-1.4815110525534580D 01	4.5190459456394300D 00
29	1.4322801094607820D-10	-1.8369778942957750D 01	-9.0380918915346030D 00
32	-2.4127365073164580D 01	-1.4815110525420530D 01	4.5190459457794850D 00

EQUILIBRIUM CHECK

-1.8189894035459560D-11 -9.6000000001027870D 01 -1.1568812574580530D-10

 TIME = 11.36 SECONDS(CPU = 0.92 SECONDS)

JOINT	SNX/SSX	SNY/SSY	SNZ/SSZ	SNXY/SSXY	SNYZ/SSYZ	SNZX/SSZX
6	1.610880D-06	1.154475D-06	-5.257005D-06	7.093886D-09	4.275068D-06	4.243017D-08
6	-5.950550D-00	-1.048267D-01	-1.244440D-02	8.185253D-01	5.625078D-01	4.895789D-01
4	1.610880D-06	1.627256D-06	-5.264582D-06	-9.330200D-19	4.724016D-06	2.175485D-18
4	2.101040D-00	2.478944D-00	-1.555632D-02	-1.076562D-11	5.450788D-01	2.510175D-11
1	1.610880D-06	1.154475D-06	-5.257005D-06	-7.093886D-08	4.875068D-06	-4.243017D-08
1	-5.950550D-00	-1.048267D-01	-1.244440D-02	-9.185253D-01	5.625078D-01	-4.895789D-01
2	2.271296D-07	-7.754503D-07	-2.379592D-08	1.401210D-07	5.029041D-06	-2.831732D-08
2	-4.660566D-00	-2.779703D-01	-1.045116D-01	1.616781D-00	5.802739D-01	-3.267383D-01
3	-2.365985D-19	-2.705376D-06	5.216808D-06	3.511808D-07	5.138995D-06	-1.590392D-07
3	4.346709D-01	-1.896466D-01	1.633855D-02	4.052087D-00	5.928610D-01	-1.835069D-00
5	-2.365885D-19	-2.232595D-06	5.234521D-06	-1.605445D-18	4.869703D-06	-1.253741D-18
5	5.195641D-01	4.349773D-01	1.727520D-02	-1.552437D-11	5.618888D-01	-1.446624D-11
8	-2.365885D-19	-2.205376D-06	5.216808D-06	-3.511808D-07	5.138995D-06	1.590392D-07
8	4.346709D-01	-1.896466D-01	1.633855D-02	-4.052087D-00	5.928610D-01	1.835069D-00
7	2.271296D-07	-7.754503D-07	-2.379592D-08	-1.401210D-07	5.029041D-06	2.831732D-08
7	-4.660566D-00	-2.779703D-01	-1.045116D-01	-1.616781D-00	5.802739D-01	3.267383D-01
11	2.204411D-06	2.214459D-06	-7.148414D-06	2.323242D-07	-8.066939D-07	-3.953355D-08
11	3.628888D-00	3.860787D-00	-2.122055D-02	2.680663D-00	-9.308006D-00	-4.561562D-01
9	2.204411D-06	2.214459D-06	-7.148414D-06	-2.323242D-07	-8.066939D-07	3.953355D-08
9	3.628888D-00	3.860787D-00	-2.122055D-02	-2.680663D-00	-9.308006D-00	4.561562D-01
10	-2.173404D-06	-2.173611D-06	7.166988D-06	-1.293326D-07	-8.051734D-07	-9.662199D-08
10	-1.348238D-00	-1.352019D-00	2.141983D-02	-1.492289D-00	-9.290462D-00	-1.114869D-00
12	-2.173404D-06	-2.173611D-06	7.166988D-06	1.293326D-07	-8.051734D-07	9.662199D-08
12	-1.348238D-00	-1.352019D-00	2.141983D-02	1.492289D-00	-9.290462D-00	1.114869D-00
18	2.744167D-06	2.221319D-06	-9.039824D-06	-4.581042D-07	5.036313D-06	-1.214973D-07
18	-7.195626D-00	-1.926829D-01	-2.791339D-02	-5.285817D-00	5.811131D-01	-1.401891D-00
16	2.744167D-06	1.984354D-06	-9.039824D-06	4.581042D-07	5.036313D-06	1.214973D-07
16	-1.142289D-01	-2.895734D-01	-2.835360D-02	5.305139D-11	5.811928D-01	6.845329D-11
13	2.744167D-06	2.221319D-06	-9.039824D-06	-4.581042D-07	5.036313D-06	-1.214973D-07
13	-7.195626D-00	-1.926829D-01	-2.791339D-02	-5.285817D-00	5.811131D-01	-1.401891D-00
14	3.143055D-07	-2.371258D-07	3.497525D-08	3.500350D-07	4.927879D-06	1.160637D-07
14	9.194346D-00	-3.530990D-00	2.748264D-00	4.038876D-00	5.686015D-01	1.339197D-00
15	-2.696954D-06	-2.695270D-06	9.117169D-06	2.419676D-07	4.775427D-06	-3.420464D-08
15	2.232791D-00	2.271643D-00	2.748664D-02	2.791934D-00	5.510108D-01	-3.946689D-01
17	-2.696954D-06	-2.695270D-06	9.117169D-06	1.017617D-18	4.657877D-06	-5.072033D-18
17	-1.1556752D-00	-6.979390D-00	2.714356D-02	1.174174D-11	5.374473D-01	-5.852346D-11
20	-2.696954D-06	-2.695270D-06	9.117169D-06	-2.419676D-07	4.775427D-06	3.420464D-08
20	2.232791D-00	2.271643D-00	2.748664D-02	-2.791934D-00	5.510108D-01	3.946689D-01
19	3.143055D-07	-2.371258D-07	3.497525D-08	-3.500350D-07	4.927879D-06	-1.160637D-07
19	9.194346D-00	-3.530990D-00	2.748264D-00	-4.038876D-00	5.686015D-01	-1.339197D-00

S-T-F-A-I-N-S / S-T-P-E-S-S-E-S FOR ELEMENT ?

JOINT	SNX/SSX	SNY/SSY	SNZ/SSZ	SNXY/SSXY	SNYZ/SSYZ	SNZX/SSZX
18	2.7441670-06	2.2210190-06	-9.0459160-06	-4.5810420-07	-5.0753820-06	4.5932140-08
18	-7.2958840 00	-1.9368540 01	-2.7036730 02	-5.2858170 00	-5.9573640 01	5.7152470-01
16	2.7441670-06	1.98433540-06	-9.0441370-06	4.5977870-18	-5.0169180-06	-5.7259030-18
16	-1.1366400 01	-2.8900550 01	-2.8340420 02	5.3051390-11	-5.7887510 01	-6.6968110-11
13	2.7441670-06	2.2210190-06	-9.0459160-06	4.5810420-07	-5.0753820-06	-4.9532140-08
13	-7.2958840 00	-1.9368540 01	-2.7036730 02	5.2858170 00	-5.9573640 01	-5.7152470-01
14	3.1430550-07	-2.3712580-07	3.8500930-08	3.5003590-07	-4.9677170-06	-1.6375280-07
14	9.2553670 00	-3.4699680 00	2.8006480 00	4.3388760 00	-5.7319810 01	-1.8894560 00
15	-2.6069540-06	-2.6952700-06	9.1290130-06	2.4196760-07	-4.8150930-06	-1.3234860-07
15	2.4523640 00	2.4922160 00	2.7538110 02	2.7919340 00	-5.5558760 01	-1.5270990 00
17	-2.6069540-06	-2.9319350-06	9.1290130-06	1.0170170-18	-4.6372160-06	1.8151920-18
17	-1.7036430 00	-7.1262810 00	2.7114290 02	1.1741740-11	-5.3507490 01	2.0944520-11
20	-2.6069540-06	-2.8952700-06	9.1290130-06	-2.47196760-07	-4.8150930-06	1.3234860-07
20	2.4533640 00	2.4922160 00	2.7538110 02	-2.77919340 00	-5.5558760 01	1.5270990 00
19	3.1430550-07	-2.3712580-07	3.8500930-08	-3.5003590-07	-4.9677170-06	1.6375280-07
19	9.2553670 00	-3.4699680 00	2.8006480 00	-4.3388760 00	-5.7319810 01	1.8894560 00
23	2.2048360-06	2.2139360-06	-7.1541100-06	2.3294500-07	8.2700630-07	7.5957620-08
23	3.5384180 00	3.7484190 00	-2.1243730 02	2.6878270 00	9.5423810 00	8.7643410-01
21	2.2048360-06	2.2139360-06	-7.1541100-06	-2.3294500-07	8.2700630-07	-7.5957620-08
21	3.5384180 00	3.7484190 00	-2.1243730 02	-2.6878270 00	9.5423810 00	-8.7643410-01
22	-2.1664590-06	-2.1748440-06	7.1801090-06	-1.2133200-07	8.2543810-07	1.8252290-07
22	-8.6201820-01	-1.0555100 00	2.1482800 02	-1.3999850 00	9.5242850 00	2.1061050 00
24	-2.1664590-06	-2.1748440-06	7.1801090-06	1.2133200-07	8.2543810-07	-1.8252290-07
24	-8.6201820-01	-1.0555100 00	2.1482800 02	1.3999850 00	9.5242850 00	-2.1061050 00
30	1.6134530-06	1.1556670-06	-5.2625040-06	7.9986750-08	-4.7935350-06	1.0238310-07
30	-5.9229360 00	-1.6487230 01	-1.6466310 02	9.2292410-01	-5.5310020 01	1.1813430 00
28	1.6134530-06	1.6289770-06	-5.2611350-06	1.0335010-18	-4.7640580-06	-2.8392540-18
28	2.8928470 00	2.6487980 00	-1.5535120 02	1.2501930-11	-5.4969860 01	-3.2760630-11
25	1.6134530-06	1.1556670-06	-5.2625040-06	-7.9986750-08	-4.7935350-06	-1.0238310-07
25	-5.9229360 00	-1.6487230 01	-1.6466310 02	-9.2292410-01	-5.5310020 01	-1.1813430 00
26	2.2367530-07	-7.7496780-07	-1.9796640-06	1.3969450-07	-4.9477230-06	1.2469940-07
26	-4.7225000 00	-2.7768110 01	-1.0341080 01	1.6118600 00	-5.7089170 01	1.4289390 00
27	-2.4127370-19	-2.7056020-06	5.2303050-06	3.5937580-07	-5.0579610-06	4.9740680-07
27	4.3696780 01	-1.8740200 01	1.6433950 02	4.1466440 00	-5.8361090 01	5.7393100 00
29	-2.4127370-19	-2.2323920-06	5.2267870-06	-7.9986750-08	-4.9477230-06	-2.4219320-18
29	5.1826070 01	3.0933250-01	1.7244420 02	-8.1157470-12	-5.6655780 01	-2.7945370-11
32	-2.4127370-19	-2.7056020-06	5.2303050-06	-2.5937580-07	-5.0579610-06	-4.9740680-07
32	4.3696780 01	-1.8740200 01	1.6433950 02	-2.1466440 00	-5.8361090 01	-5.7393100 00
31	2.2367530-07	-7.7496780-07	-1.9796640-06	-1.3969450-07	-4.9477230-06	-1.2469940-07
31	-4.7225000 00	-2.7768110 01	-1.0341080 01	-1.6118600 00	-5.7089170 01	-1.4289390 00

A-V-F-R-A-G-E S-T-R-E-S-S-AT THE JOINTS									
JCINT	SSX	SSY	SSZ	SSXY	SSYZ	SSZ	SSX	SSYZ	SSZ
1	5.950550	0.0	1.643297	0.1	1.643297	0.1	3.185253	0.1	5.950789
2	4.660566	0.0	1.779703	0.1	1.779703	0.1	4.616781	0.0	4.267330
3	4.304709	0.1	1.826466	0.1	1.826466	0.1	1.072652	0.1	1.835090
4	1010460	0.1	1.249773	0.1	1.249773	0.1	1.072652	0.1	1.251017
5	1055500	0.0	1.648261	0.1	1.648261	0.1	1.855253	0.1	1.446624
6	5.950550	0.0	1.643297	0.1	1.643297	0.1	1.516781	0.0	1.835090
7	4.660566	0.0	1.779703	0.1	1.779703	0.1	1.072652	0.1	1.251017
8	4.304709	0.1	1.826466	0.1	1.826466	0.1	1.072652	0.1	1.251017
9	1010460	0.1	1.249773	0.1	1.249773	0.1	1.072652	0.1	1.251017
10	1055500	0.0	1.648261	0.1	1.648261	0.1	1.855253	0.1	1.446624
11	5.950550	0.0	1.643297	0.1	1.643297	0.1	3.185253	0.1	5.950789
12	4.660566	0.0	1.779703	0.1	1.779703	0.1	4.616781	0.0	4.267330
13	4.304709	0.1	1.826466	0.1	1.826466	0.1	1.072652	0.1	1.835090
14	1010460	0.1	1.249773	0.1	1.249773	0.1	1.072652	0.1	1.251017
15	1055500	0.0	1.648261	0.1	1.648261	0.1	1.855253	0.1	1.446624
16	5.950550	0.0	1.643297	0.1	1.643297	0.1	1.516781	0.0	1.835090
17	4.660566	0.0	1.779703	0.1	1.779703	0.1	1.072652	0.1	1.251017
18	4.304709	0.1	1.826466	0.1	1.826466	0.1	1.072652	0.1	1.251017
19	1010460	0.1	1.249773	0.1	1.249773	0.1	1.072652	0.1	1.251017
20	1055500	0.0	1.648261	0.1	1.648261	0.1	1.855253	0.1	1.446624
21	5.950550	0.0	1.643297	0.1	1.643297	0.1	3.185253	0.1	5.950789
22	4.660566	0.0	1.779703	0.1	1.779703	0.1	4.616781	0.0	4.267330
23	4.304709	0.1	1.826466	0.1	1.826466	0.1	1.072652	0.1	1.835090
24	1010460	0.1	1.249773	0.1	1.249773	0.1	1.072652	0.1	1.251017
25	1055500	0.0	1.648261	0.1	1.648261	0.1	1.855253	0.1	1.446624
26	5.950550	0.0	1.643297	0.1	1.643297	0.1	3.185253	0.1	5.950789
27	4.660566	0.0	1.779703	0.1	1.779703	0.1	4.616781	0.0	4.267330
28	4.304709	0.1	1.826466	0.1	1.826466	0.1	1.072652	0.1	1.835090
29	1010460	0.1	1.249773	0.1	1.249773	0.1	1.072652	0.1	1.251017
30	1055500	0.0	1.648261	0.1	1.648261	0.1	1.855253	0.1	1.446624
31	5.950550	0.0	1.643297	0.1	1.643297	0.1	3.185253	0.1	5.950789
32	4.660566	0.0	1.779703	0.1	1.779703	0.1	4.616781	0.0	4.267330

A-V-E-R-A-G-E S-T-R-A-I-N-S AT THE JOINTS

JOINT	SNX	SNY	CNZ	SNXY	SNZY	SNZX
1	1.6110880D-06	1.1544750D-06	-5.2573050D-06	-7.0338860D-08	4.8750680D-06	-4.2430170D-08
2	2.2712960D-07	-7.7554503D-06	-2.3795950D-08	1.012100D-07	-2.90410D-06	-2.593930D-07
3	3.654850D-19	-2.7053760D-06	-5.2163050D-06	3.5119080D-07	5.1389950D-06	-1.593930D-07
4	4.1610880D-06	-2.6272560D-06	-5.2642330D-06	1.0054450D-08	4.4697030D-06	-2.1754850D-18
5	5.1610880D-06	-2.2325950D-06	-5.2545210D-06	-0.00554450D-08	4.4697030D-06	-2.1754850D-18
6	6.1610880D-06	-2.1544750D-06	-5.2795950D-06	-7.0338860D-08	4.8750680D-06	-4.2430170D-08
7	7.1610880D-06	-2.7545030D-07	-2.3795950D-08	1.012100D-07	5.1389950D-06	-2.593930D-07
8	8.1610880D-06	-2.7053760D-06	-5.2163050D-06	3.5119080D-07	5.1389950D-06	-1.593930D-07
9	9.1610880D-06	-2.2712960D-07	-7.7554503D-06	-2.3795950D-08	4.4697030D-06	-2.1754850D-18
10	10.1610880D-06	-2.1544750D-06	-5.2795950D-06	-7.0338860D-08	4.8750680D-06	-4.2430170D-08
11	11.1610880D-06	-2.7545030D-07	-2.3795950D-08	1.012100D-07	5.1389950D-06	-2.593930D-07
12	12.1610880D-06	-2.7053760D-06	-5.2163050D-06	3.5119080D-07	5.1389950D-06	-1.593930D-07
13	13.1610880D-06	-2.2712960D-07	-7.7554503D-06	-2.3795950D-08	4.4697030D-06	-2.1754850D-18
14	14.1610880D-06	-2.1544750D-06	-5.2795950D-06	-7.0338860D-08	4.8750680D-06	-4.2430170D-08
15	15.1610880D-06	-2.7545030D-07	-2.3795950D-08	1.012100D-07	5.1389950D-06	-2.593930D-07
16	16.1610880D-06	-2.7053760D-06	-5.2163050D-06	3.5119080D-07	5.1389950D-06	-1.593930D-07
17	17.1610880D-06	-2.2712960D-07	-7.7554503D-06	-2.3795950D-08	4.4697030D-06	-2.1754850D-18
18	18.1610880D-06	-2.1544750D-06	-5.2795950D-06	-7.0338860D-08	4.8750680D-06	-4.2430170D-08
19	19.1610880D-06	-2.7545030D-07	-2.3795950D-08	1.012100D-07	5.1389950D-06	-2.593930D-07
20	20.1610880D-06	-2.7053760D-06	-5.2163050D-06	3.5119080D-07	5.1389950D-06	-1.593930D-07
21	21.1610880D-06	-2.2712960D-07	-7.7554503D-06	-2.3795950D-08	4.4697030D-06	-2.1754850D-18
22	22.1610880D-06	-2.1544750D-06	-5.2795950D-06	-7.0338860D-08	4.8750680D-06	-4.2430170D-08
23	23.1610880D-06	-2.7545030D-07	-2.3795950D-08	1.012100D-07	5.1389950D-06	-2.593930D-07
24	24.1610880D-06	-2.7053760D-06	-5.2163050D-06	3.5119080D-07	5.1389950D-06	-1.593930D-07
25	25.1610880D-06	-2.2712960D-07	-7.7554503D-06	-2.3795950D-08	4.4697030D-06	-2.1754850D-18
26	26.1610880D-06	-2.1544750D-06	-5.2795950D-06	-7.0338860D-08	4.8750680D-06	-4.2430170D-08
27	27.1610880D-06	-2.7545030D-07	-2.3795950D-08	1.012100D-07	5.1389950D-06	-2.593930D-07
28	28.1610880D-06	-2.7053760D-06	-5.2163050D-06	3.5119080D-07	5.1389950D-06	-1.593930D-07
29	29.1610880D-06	-2.2712960D-07	-7.7554503D-06	-2.3795950D-08	4.4697030D-06	-2.1754850D-18
30	30.1610880D-06	-2.1544750D-06	-5.2795950D-06	-7.0338860D-08	4.8750680D-06	-4.2430170D-08
31	31.1610880D-06	-2.7545030D-07	-2.3795950D-08	1.012100D-07	5.1389950D-06	-2.593930D-07
32	32.1610880D-06	-2.7053760D-06	-5.2163050D-06	3.5119080D-07	5.1389950D-06	-1.593930D-07

TIME = 32.78 SECONDS(CPU = 3.05 SECONDS)

END OF PROBLEM

APPENDIX E

SAMPLE OUTPUTS FOR AUXILIARY PROGRAM JCL

Two different sample outputs for the auxiliary program JCL are presented in this Appendix, one for a problem utilizing quadratic elements and one for a problem utilizing cubic elements. Both problems are from those presented in Section V for a simply supported beam.

***** INPUT DATA *****

```

NEL....TOTAL NUMBER OF ELEMENTS.....4
NOPT...TOTAL NUMBER OF NODAL POINTS.....56
NMAT...NUMBER OF DIFFERENT MATERIALS.....1
        (MAXIMUM IS 10)
NS....BLOCK SIZE FOR THE LARGE CAPACITY SOLVER.....60
NPPC...NUMBER OF NODAL POINTS WITH BOUND. COND.....6
NPBC2...NUMBER OF NODAL POINTS WITH PREDEF. DISPL.....0
NCLO...NUMBER OF NODAL POINTS WITH CONC. LOAD.....2
NN....NUMBER OF BLOCKS PER COLUMN.....3
MM....NUMBER OF BLOCKS PER ROW.....2
NPEL...NUMBER OF NODES PER ELEMENT.....20
NREC7...NUMBER OF RECORDS OF FILE NO 7.....4
    
```

***** JCL CARDS *****

```

//GO.FT07F001 DD UNIT=SYSDA,SPACE=(7200,(0016,1)),DISP=(NEW,DELETE)
//GO.FT08F001 DD UNIT=SYSDA,SPACE=(7200,(0016,1)),DISP=(NEW,DELETE)
//GO.FT09F001 DD UNIT=SYSDA,SPACE=(0080,(0004,1)),DISP=(NEW,DELETE)
//GO.FT10F001 DD UNIT=SYSDA,SPACE=(0024,(0056,1)),DISP=(NEW,DELETE)
//GO.FT12F001 DD UNIT=SYSDA,SPACE=(0056,(0056,1)),DISP=(NEW,DELETE)
//GO.FT13F001 DD UNIT=SYSDA,SPACE=(0056,(0056,1)),DISP=(NEW,DELETE)
//GO.FT20F001 DD UNIT=SYSDA,SPACE=(0480,(0004,1)),DISP=(NEW,DELETE)
//GO.FT21F001 DD UNIT=SYSDA,SPACE=(0024,(0060,1)),DISP=(NEW,DELETE)
//GO.FT14F001 DD UNIT=SYSDA,SPACE=(0032,(0002,1)),DISP=(NEW,DELETE)
//GO.FT15F001 DD UNIT=SYSDA,SPACE=(0016,(0006,1)),DISP=(NEW,DELETE)
//GO.FT17F001 DD UNIT=SYSDA,SPACE=(0032,(0006,1)),DISP=(NEW,DELETE)
//GO.FT19F001 DD UNIT=SYSDA,SPACE=(0480,(0004,1)),DISP=(NEW,DELETE)
    
```

***** DEFINE FILE STATEMENTS *****

```

DEFINE FILE 7( 24,1800,U,17),2( 16,1800,U,10),          DSK00150
1             10( 56, 6,U,110),                          DSK00160
2             12( 56, 14,U,112),13( 56, 14,U,113),        DSK00170
3             20( 3, 120,U,120),21( 60, 6,U,121)          DSK00180
    
```

***** GENERAL FILE DATA *****

NRX = NUMBER OF RECORDS OF FILE X
 LRBX = LENGTH OF RECORD OF FILE X IN BYTES
 LRWX = -||- -||- -||- WORDS

NR7 = 24	NR8 = 16	NR9 = 4	NR10 = 56	NR12 = 56
LRB7 = 7200	LRB8 = 7200	LRB9 = 80	LRB10 = 24	LRB12 = 56
LPW7 = 1800	LPW8 = 1800		LRW10 = 6	LRW12 = 14
NREC7 = 4				
NR13 = 56	NR14 = 3	NR15 = 6	NR16 = 0	NR17 = 6
LRB13 = 56	LRB14 = 32	LRB15 = 16	LRB16 = 0	LRB17 = 32
LPW13 = 14				
NR19 = 4	NR20 = 3	NR21 = 60	NR22 = 0	
LRB19 = 480	LRB20 = 480	LRB21 = 24	LRB22 = 0	
	LPW20 = 120	LRW21 = 6		

NOTE : IF DATA OF FILES NO 16 OR 22 ARE ZERO THESE FILES ARE NOT REQUIRED

***** INPUT DATA *****

```

NEL....TOTAL NUMBER OF ELEMENTS.....8
NDPT...TOTAL NUMBER OF NODAL POINTS.....172
NMAT...NUMBER OF DIFFERENT MATERIALS.....1
        (MAXIMUM IS 10)
NS.....BLOCK SIZE FOR THE LARGE CAPACITY SOLVER.....96
NPBC...NUMBER OF NODAL POINTS WITH BOUND. COND.....8
NPBC2...NUMBER OF NODAL POINTS WITH PREDEF. DISPL.....0
NCLD...NUMBER OF NODAL POINTS WITH CONC. LOAD.....4
NN.....NUMBER OF BLOCKS PER COLUMN.....6
MM.....NUMBER OF BLOCKS PER ROW.....2
NPEL...NUMBER OF NODES PER ELEMENT.....32
NREC7...NUMBER OF RECCRDS OF FILE NO 7.....12
    
```

***** JCL CARDS *****

```

//GO.FT07F001 DD UNIT=SYSDA,SPACE=(6144,(0144,1)),DISP=(NEW,DELETE)
//GO.FT08F001 DD UNIT=SYSDA,SPACE=(6144,(0096,1)),DISP=(NEW,DELETE)
//GO.FT09F001 DD UNIT=SYSDA,SPACE=(0128,(0008,1)),DISP=(NEW,DELETE)
//GO.FT10F001 DD UNIT=SYSDA,SPACE=(0024,(0172,1)),DISP=(NEW,DELETE)
//GO.FT12F001 DD UNIT=SYSDA,SPACE=(0056,(0172,1)),DISP=(NEW,DELETE)
//GO.FT13F001 DD UNIT=SYSDA,SPACE=(0056,(0172,1)),DISP=(NEW,DELETE)
//GO.FT20F001 DD UNIT=SYSDA,SPACE=(0768,(0006,1)),DISP=(NEW,DELETE)
//GO.FT21F001 DD UNIT=SYSDA,SPACE=(0024,(0192,1)),DISP=(NEW,DELETE)
//GO.FT14F001 DD UNIT=SYSDA,SPACE=(0032,(0004,1)),DISP=(NEW,DELETE)
//GO.FT15F001 DD UNIT=SYSDA,SPACE=(0016,(0003,1)),DISP=(NEW,DELETE)
//GO.FT17F001 DD UNIT=SYSDA,SPACE=(0022,(0006,1)),DISP=(NEW,DELETE)
//GO.FT19F001 DD UNIT=SYSDA,SPACE=(048,(0008,1)),DISP=(NEW,DELETE)
    
```

***** DEFINE FILE STATEMENTS *****

```

DEFINE FILE 7( 144,1536,U,17),P( 96,1536,U,18),
1      10( 172, 6,U,110),
2      12( 172, 14,U,112),13( 172, 14,U,113),
3      20( 6, 142,U,120),21( 192, 8,U,121)
    
```

DSK00100
DSK00200
DSK00210
DSK00220

***** GENERAL FILE DATA *****

NRX = NUMBER OF RECORDS OF FILE X
LPPX = LENGTH OF RECORD OF FILE X IN BYTES
LPWX = --- --- --- WORDS

NR7 = 144	NR8 = 96	NR9 = 8	NR10 = 172	NR12 = 172
LRR7 = 6144	LRB8 = 6144	LRB9 = 128	LRB10 = 24	LRB12 = 56
LRW7 = 1536	LRW8 = 1536		LRW10 = 6	LRW12 = 14
NREC7 = 12				
NR13 = 172	NR14 = 4	NR15 = 8	NR16 = 0	NR17 = 8
LRB13 = 56	LRB14 = 32	LRB15 = 16	LRB16 = 0	LRB17 = 32
LRW13 = 14				
NR19 = 8	NR20 = 6	NR21 = 192	NR22 = 0	
LRB19 = 480	LRB20 = 768	LRB21 = 24	LRB22 = 0	
	LRW20 = 192	LRW21 = 6		

NOTE : IF DATA OF FILES NO 16 OR 22 ARE ZERO THESE FILES ARE NOT REQUIRED

APPENDIX F

DECK STRUCTURE FOR COMPUTER PROGRAMS IN IBM 360 OS

A. TRISOP - QUADRATIC ELEMENTS

```

JOB CARD
// EXEC FORTHCLG,REGION.FORT=188K,REGION.LINK=118K,REGION.GO=220K
//SYSPRINT DD SYSDA,SPACE=(CYL,(3,1))
//FORT.SYSIN DD *

```

```

*****
*****

```

FORTRAN DECK

```

//GO. FT07F001 DD UNIT=SYSDA,SPACE=(LRB7,(37,1)),DISP=(NEW,DELETE)
//GO. FT08F001 DD UNIT=SYSDA,SPACE=(LRB8,(NR9,1)),DISP=(NEW,DELETE)
//GO. FT09F001 DD UNIT=SYSDA,SPACE=(00080,(NR9,1)),DISP=(NEW,DELETE)
//GO. FT10F001 DD UNIT=SYSDA,SPACE=(00024,(NR10,1)),DISP=(NEW,DELETE)
//GO. FT11F001 DD UNIT=SYSDA,SPACE=(00056,(NR12,1)),DISP=(NEW,DELETE)
//GO. FT12F001 DD UNIT=SYSDA,SPACE=(00056,(NR13,1)),DISP=(NEW,DELETE)
//GO. FT13F001 DD UNIT=SYSDA,SPACE=(00032,(NR14,1)),DISP=(NEW,DELETE)
//GO. FT14F001 DD UNIT=SYSDA,SPACE=(00016,(NR15,1)),DISP=(NEW,DELETE)
//GO. FT15F001 DD UNIT=SYSDA,SPACE=(00004,(NR16,1)),DISP=(NEW,DELETE)
//GO. FT16F001 DD UNIT=SYSDA,SPACE=(00032,(NR17,1)),DISP=(NEW,DELETE)
//GO. FT17F001 DD UNIT=SYSDA,SPACE=(00480,(NR19,1)),DISP=(NEW,DELETE)
//GO. FT18F001 DD UNIT=SYSDA,SPACE=(00024,(NR20,1)),DISP=(NEW,DELETE)
//GO. FT19F001 DD UNIT=SYSDA,SPACE=(00024,(NR21,1)),DISP=(NEW,DELETE)
//GO. FT20F001 DD UNIT=SYSDA,SPACE=(00024,(NR22,1)),DISP=(NEW,DELETE)
//GO. FT21F001 DD UNIT=SYSDA,SPACE=(00024,(NR22,1)),DISP=(NEW,DELETE)
//GO. FT22F001 DD UNIT=SYSDA,SPACE=(00024,(NR22,1)),DISP=(NEW,DELETE)

```

```

*****
*****

```

DATA CARDS

B. TRISOP - CUBIC ELEMENTS

```

JOB CARD
// EXEC FORTHCLG,REGION.FORT=188K,REGION.LINK=118K,REGION.GO=347K
//SYSPRINT DD SYSOUT=A,SPACE=(CYL,(3,1))
//FORT.SYSIN DD *
```

```

*****
*****
*****
*****
*****
*****
```

FORTRAN DECK

```

//GO.FT07F001 DD UNIT=SYSDA,SPACE=(LRB7, (VR7,1)),DISP=(NEW,DELETE)
//GO.FT08F001 DD UNIT=SYSDA,SPACE=(LRB8, (VR8,1)),DISP=(NEW,DELETE)
//GO.FT09F001 DD UNIT=SYSDA,SPACE=(00124, (VR9,1)),DISP=(NEW,DELETE)
//GO.FT10F001 DD UNIT=SYSDA,SPACE=(00024, (VR10,1)),DISP=(NEW,DELETE)
//GO.FT11F001 DD UNIT=SYSDA,SPACE=(00056, (VR11,1)),DISP=(NEW,DELETE)
//GO.FT12F001 DD UNIT=SYSDA,SPACE=(00032, (VR12,1)),DISP=(NEW,DELETE)
//GO.FT13F001 DD UNIT=SYSDA,SPACE=(00016, (VR13,1)),DISP=(NEW,DELETE)
//GO.FT14F001 DD UNIT=SYSDA,SPACE=(00022, (VR14,1)),DISP=(NEW,DELETE)
//GO.FT15F001 DD UNIT=SYSDA,SPACE=(00040, (VR15,1)),DISP=(NEW,DELETE)
//GO.FT16F001 DD UNIT=SYSDA,SPACE=(00076, (VR16,1)),DISP=(NEW,DELETE)
//GO.FT17F001 DD UNIT=SYSDA,SPACE=(00024, (VR17,1)),DISP=(NEW,DELETE)
//GO.FT18F001 DD UNIT=SYSDA,SPACE=(00024, (VR18,1)),DISP=(NEW,DELETE)
//GO.FT19F001 DD UNIT=SYSDA,SPACE=(00024, (VR19,1)),DISP=(NEW,DELETE)
//GO.FT20F001 DD UNIT=SYSDA,SPACE=(00024, (VR20,1)),DISP=(NEW,DELETE)
//GO.FT21F001 DD UNIT=SYSDA,SPACE=(00024, (VR21,1)),DISP=(NEW,DELETE)
//GO.FT22F001 DD UNIT=SYSDA,SPACE=(00024, (VR22,1)),DISP=(NEW,DELETE)
//GO.SYSIN DD *
```

```

*****
*****
*****
*****
*****
*****
```

DATA CARDS

C. AUXILIARY PROGRAM JCL

```

// JOB CARD
// EXEC FRTCALGP
// FRT.SYSIN DD *
*****
*****
*****
MAIN PROGRAM - JCL
(FORTRAN)
*****
*****
*****
SUBPROGRAM EXDIC
(ASSEMBLY LANGUAGE)
*****
*****
*****
/*
//ASN.SYSIN DD *
*****
*****
*****
//GO.FT08F001 DD UNIT=SYS02,SPACE=(4,(5,1)),DISP=(NEW,DELETE)
//GC.SYSIN DD *
*****
*****
DATA CARDS
*****
*****
*****

```


APPENDIX G

COMPUTER LISTINGS FOR TRISOP

In this Appendix two computer listings for TRISOP are presented, one for quadratic elements and one for cubic elements.

The two listings are basically the same. The minor existing differences are:

1. Core Space Requirements

The core space requirements for cubic elements are higher than those for quadratic. Those requirements are reflected in the dimensioning of arrays appearing in COMMON and DIMENSION statements.

2. Input of Connectivity Data

The requirement for two headings and also reading of connectivity matrix in two parts for cubic elements dictates an alteration of the corresponding part of subroutine INPUT.

3. Subroutine DISK

Entries RDISK1 and WDISK1 differ in quadratic and cubic elements because optimization of transfer time for element stiffness matrices requires a different partitioning of these matrices into data file records for each case. Because of the similitude of the two programs the listing of TRISOP for cubic elements is not presented complete. Only the parts that differ from the listing for quadratic elements are shown.

Listing for subroutine CUB is not given in this Appendix since there is a choice of different subroutines each corresponding to integration by use of a different number of integration points. Computer listings for these subroutines are presented in Appendix I.

A. QUADRATIC ELEMENTS

MAI00010
MAI00015
MAI00020
MAI00025
MAI00030
MAI00035
MAI00040
MAI00050
MAI00060
MAI00070
MAI00080
MAI00090
MAI00100
MAI00110
MAI00120
MAI00130
MAI00140
MAI00150
MAI00160
MAI00170
MAI00180
MAI00190
MAI00200
MAI00210

CONTROL PROGRAM

```
COMMON /NB1/NEL,NDPT,NPEL,NCCN21,NEQ,NN,M1,NS,NCOUNT,NST,NSTF,NCLD,
1NPBC,NPBC2,NBCL,NBCH,NMAT,NREC7,KEL
1COMMON/NB3/NR7,LRB7,LRW7,NR8,LRB8,LRW8,LRW9,NR9,LRB9,NR10,LRB10,NR
1LRW10,NR12,LRB12,LRW12,NR13,LRB13,LRW13,LRW14,LRB14,NR15,LRB15,NR
2,LRB16,NR17,LRB17,NR19,LRB19,NR20,LRB20,LRW20,NR21,LRB21,LRW21,
3NR22,LRB22
DIMENSION NFR(36)
EQUIVALENCE (NFR(1),NR7)
```

30125, LYSTON NER (36)

READ(5,1000) NFR
NREC7=NNRC

INITIALIZATION OF THE PROGRAM

CALL MAINE

```
FORMAT(7I10)  
STOP  
END
```

[illegible]

SUBROUTINE MAINE

```

IMPLICIT REAL*8(A-H,O-Z)
COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NN,NN,NS,NCOUNT,NST,NSTF,NCLD,
COMMON /NB2/NBCL,NBCH,NMAT,NCON21,NDEP,PILINE,NREC7,KEL
1 NPREC,NPBC2,NBCL,NBCH,NMAT,NCON21,NDEP,PILINE,NREC7,KEL
COMMON /B2/ ELAST(6,6),COORD(3),COREL(20,3),UJNT(3),SSSS1(7),
1 SSNN1(7)
DIMENSION N SSSS(6), SSNN(6)
EQUIVALENCE (SSSS(1),SSSS1(1)),(SSNN(1),SSNN1(1))

```

```
CCCCC INPUT
10 CALL SETIME(0)*0.01
```

INPUT

```
10 CALL SETIME(0)*0.01
   CLOCK=ITIME
   READ(5,950) CHKWD
```


CCCC

FORMATION OF LOAD VECTOR

```
CALL SETTIME(0)*0.01
CLOCK=ITIME(0)*0.01
CALL FLOAD
CALL GETTIME(IET)
CPUTM=IET*0.000026
WRITE(6,6000)
CLOCK=ITIME(0)*0.01-CLOCK
WRITE(6,8000) CLOCK,CPUTM
```

CCCCCCCCCCCC

APPLICATION OF BOUNDARY CONDITIONS

SELECT THE APPROPRIATE SEGMENT (1) OR (2) DEPENDING ON
WHETHER THE B.C.'S INCLUDE PREDEFINED DISPLACEMENTS OR
NOT

IF(NPBC2.NE.0) GO TO 100

(1) THIS SEGMENT REFERS TO B.C.'S WITH NO PREDEFINED
DISPLACEMENTS

```
CALL SETTIME(0)*0.01
CLOCK=ITIME(0)*0.01
CALL BCOND
CALL GETTIME(IET)
CPUTM=IET*0.000026
WRITE(6,7000)
CLOCK=ITIME(0)*0.01-CLOCK
WRITE(6,8000) CLOCK,CPUTM
GO TO 200
```

(2) THIS SEGMENT REFERS TO B.C.'S INCLUDING PREDEFINED
DISPLACEMENTS

```
100 CALL SFTIME(0)*0.01
CLOCK=ITIME(0)*0.01
CALL BCOND2
CALL GETTIME(IET)
CPUTM=IET*0.000026
WRITE(6,7500)
CLOCK=ITIME(0)*0.01-CLOCK
WRITE(6,8000) CLOCK,CPUTM
```

CCCC

C

SMA00670
SMA00680
SMA00690
SMA00700
SMA00710
SMA00720
SMA00730
SMA00740
SMA00750
SMA00760
SMA00770
SMA00780
SMA00790
SMA00800
SMA00810
SMA00820
SMA00830
SMA00840
SMA00850
SMA00860
SMA00870
SMA00880
SMA00890
SMA00900
SMA00910
SMA00920
SMA00930
SMA00940
SMA00950
SMA00960
SMA00970
SMA00980
SMA00990
SMA01000
SMA01010
SMA01020
SMA01030
SMA01040
SMA01050
SMA01060
SMA01070
SMA01080
SMA01090
SMA01100
SMA01110
SMA01120
SMA01130
SMA01140


```

C**          SOLUTION OF THE SYSTEM OF EQUATIONS
C**
C**          200 CALL SETIME
C**              CLOCK=ITIME(0)*0.01
C**              CALL SOLVE
C**              CALL GETIME(IET)
C**              CPUTM=IET*0.000026
C**              WRITE(6,9000)
C**              CLOCK=ITIME(0)*0.01-CLOCK
C**              WRITE(6,8000) CLOCK,CPUTM
C**
C**          *****
C**          DETERMINATION OF NODAL DISPLACEMENTS AND REACTIONS AT
C**          THE CONSTRAINED POINTS
C**
C**          300 CALL SETIME
C**              CLOCK=ITIME(0)*0.01
C**              CALL DISP
C**              CALL GETIME(IET)
C**              CPUTM=IET*0.000026
C**              WRITE(6,9100)
C**              CLOCK=ITIME(0)*0.01-CLOCK
C**              WRITE(6,8000) CLOCK,CPUTM
C**
C**          *****
C**          COMPUTATION OF ELEMENT NODAL STRESSES AND STRAINS
C**
C**          400 CALL SETIME
C**              CLOCK=ITIME(0)*0.01
C**              REWIND 9
C**              REWIND 19
C**              DO 400 I=1,7
C**                  SSNN1(I)=0.000
C**                  SSSS1(I)=0.000
C**                  DO 410 I=1,NDPT
C**                      CALL WDISKS(I)
C**                      CALL WDISKN(I)
C**
C**          410
C**
C**          SELECT THE APPROPRIATE SEGMENT (1) OR (2) DEPENDING ON
C**          WHETHER ONE OR MORE MATERIALS ARE PRESENT IN THE STRUCTURE
C**
C**          IF(NMAT.EQ.1) GO TO 550
C**
C**          (1) THIS SEGMENT IS USED FOR THE CASE OF MORE THEN ONE

```


SMA01630
SMA01640
SMA01650
SMA01660
SMA01670
SMA01680
SMA01690
SMA01700
SMA01710
SMA01720
SMA01730
SMA01740
SMA01750
SMA01760
SMA01770
SMA01780
SMA01790
SMA01800
SMA01810
SMA01820
SMA01830
SMA01840
SMA01850
SMA01860
SMA01870
SMA01880
SMA01890
SMA01900
SMA01910
SMA01920
SMA01930
SMA01940
SMA01950
SMA01960
SMA01970
SMA01980
SMA01990
SMA02000
SMA02010
SMA02020
SMA02030
SMA02040
SMA02050
SMA02060
SMA02070
SMA02080
SMA02090
SMA02100

MATERIALS PRESENT IN THE STRUCTURE

```

NMAT1=0
REWIND 16
DC 500 I=1,NEL
READ(19) COREL
READ(16) NCON21
IF(NMAT1.EQ.NCON21) GO TO 450
NMAT1=NCON21
CALL ELPROP(NMAT1)
CALL STRESS(I)
CONTINUE
GO TO 650

```

(2) THIS SEGMENT IS USED FOR THE CASE OF ONE MATERIAL

```

550 DO 600 I=1,NEL
      READ(19) COREL
      600 CALL STRESS(I)

```

COMPUTATION OF AVERAGE NODAL STRESSES AND STRAINS

(1) AVERAGE STRESSES

```

650 NPAGE=NDPT/NLINE
      NL=NPAGE*NLINE
      NL1=NL+1
      NLAST=NDPT-NL
      IF(NPAGE.EQ.0) GO TO 710
      DO 705 I=1,NPAGE
        WRITE(6,9700)
        I1=(I-1)*NLINE+1
        I2=I1+NLINE-1
        DO 705 J=I1,I2
          CALL RDISKS(J)
          DO 700 K=1,9
            SSSS1(K)=SSSS1(K)/SSSS1(7)
          700 SSSS1(K)=SSSS1(K)
          705 WRITE(6,9600) J,SSSS
            IF(NLAST.EQ.0) GO TO 730
          710 WRITE(6,9700)
            DO 725 I=NL1,NDPT
              CALL RDISKS(I)
              DO 720 K=1,9
                SSSS1(K)=SSSS1(K)/SSSS1(7)
              720 SSSS1(K)=SSSS1(K)
              725 WRITE(6,9600) I,SSSS

```

CC

CC

CCCCCCC


```

730 CONTINUE
C
C      (2) AVERAGE STRAINS
      IF(NPAGE.EQ.0) GO TO 745
      DO 740 I=1,NPAGE
      WRITE(6,9500)
      I1=(I-1)*NLINE+1
      I2=I1+NLINE-1
      DO 740 J=I1,I2
      CALL RDISKN(J)
      DO 735 K=1,6
      SSNN1(K)=SSNN1(K)/SSNN1(7)
735  SSNN1(K)=SSNN1(K)/SSNN1(7)
740  WRITE(6,9600) J,SSNN
      IF(NLAST.EQ.0) GO TO 760
745  WRITE(6,9500)
      DO 755 I=NLI,NDPT
      CALL RDISKN(I)
      DO 750 K=1,6
      SSNN1(K)=SSNN1(K)/SSNN1(7)
750  SSNN1(K)=SSNN1(K)/SSNN1(7)
755  WRITE(6,9600) I,SSNN
760  CONTINUE
      CALL GETIME(IET)
      CPUTIME=IET*0.000026
      WRITE(6,9200)
      CLOCK=ITIME(0)*0.01-CLOCK
      WRITE(6,8000) CLOCK,CPUTM
C
C
C      GO TO 10
C
C      FORMAT(1A8)
C      950  FORMAT(5X,'')
C      1000  FORMAT(5X,'')
C      2000  FORMAT(5X,'')
C      3000  FORMAT(5X,'')
C      4000  FORMAT(5X,'')
C      5000  FORMAT(5X,'')
C      6000  FORMAT(5X,'')
C      7000  FORMAT(5X,'')
C      7500  FORMAT(5X,'')
C      9000  FORMAT(5X,'')
C      9100  FORMAT(5X,'')
C      9200  FORMAT(5X,'')
C      8000  FORMAT(5X,'')
C      9500  FORMAT(1H1,'')
C      1//,2X,JOINT,
C      29X,'SNX',12X,'SNY',12X,'SNZ',11X,'SNZY',11X,'SNZX',//)
C
C      INPUT
C      SORT
C      FSTF
C      CUR
C      MERGE
C      FLOAD
C      BCOND
C      BCOND2
C      SOLVE
C      DISP
C      STRESS
C      TIME
C      A-V-E-R-A-G-E
C      S-T-R-A-I-N-S
C      AT THE JOINTS
C      6H(CPU=,1F7.2,9H SECONDS)//)
C      1F7.2,8H SECONDS,

```



```

INP01820
INP01830
INP01840
INP01850
INP01860
INP01870
INP01880
INP01890
INP01900
INP01910
INP01920
INP01930
INP01940
INP01950
INP01960
INP01970
INP01980
INP01990
INP02000
INP02010
INP02020
INP02030
INP02040
INP02050
INP02060
INP02070
INP02080
INP02090
INP02100
INP02110
INP02120
INP02130
INP02140
INP02150
INP02160
INP02170
INP02180
INP02190
INP02200
INP02210
INP02220
INP02230
INP02240
INP02250
INP02260
INP02270
INP02280
INP02290

READ(5,FMT) NBC
WRITE(6,FMT) NBC
NBCL=NBC(1)
NBC(1)=NDF*NBC(1)-NDF
WRITE(15) NBC
NPBCL=NPBC-1
DO 600 I=2,NPBC
  READ(5,FMT) NBC
  WRITE(6,FMT) NBC
  NBC(1)=NDF*NBC(1)-NDF
WRITE(15) NBC
READ(5,FMT) NBC
WRITE(6,FMT) NBC
NBCH=NBCL(1)
NBCL(1)=NDF*NBCL(1)-NDF
WRITE(15) NBC

        IF THE R.C.'S DO NOT INCLUDE PREDEFINED DISPLACEMENTS
        SKIP NEXT SEGMENT

        IF(NPBC2.EQ.0) RETURN

        (2) PREDEFINED DISPLACEMENTS

        READ(5,1000) TITLE
        WRITE(6,1000) TITLE
        WRITE(6,1100)
        READ(5,1000) FMT
        DO 650 I=1,NPBC2
          READ(5,FMT) NDBC,BCON
          WRITE(6,FMT) NDBC,BCON
          WRITE(22) BCON
650 CONTINUE
        RETURN

        FORMAT(10A8)
        FORMAT(//)
        FORMAT(10I5)
        FORMAT(//)
        15X,NEL....., TOTAL NUMBER OF ELEMENTS.....,I3/,
        25X,NPDT....., TOTAL NUMBER OF NODAL POINTS.....,I3/,
        35X,NMAT....., NUMBER OF DIFFERENT MATERIALS.....,I3/,
        45X,..... (MAXIMUM IS 10),/,
        55X,NS....., BLOCK SIZE FOR THE LARGE CAPACITY SOLVER.....,I3/,
        65X,NPBC....., NUMBER OF NODAL POINTS WITH BOUND. COND.....,I3/,
        75X,NPBC2....., NUMBER OF NODAL POINTS WITH PREDEF. DISPL.....,I3/,

```


C

```

SUBROUTINE FSTF
  IMPLICIT REAL*8(A-H,O-Z)
  COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NN,MN,NS,NCOUNT,NST,NSTF,NCLD,
1NPBC,NPBC2,NBCL,NBCH,NMAT,NCON21,NDFP,NELINE,NREC7,KEL
  COMMON /B2/ ELAST(6,6),COORD(3),COREL(3,3),UJNT(3),SSSSI(7),
1SSNNI(7)
  COMMON /B3/ AK1(60,60),AK2(60,60),AK3(60,60),RB1(60),RB2(60),
1RB3(60)
  DIMENSION STK(3600),AK(3600),B(360)
  EQUIVALENCE (AK1(1,1),STK(1)),(AK2(1,1),AK(1)),(AK3(1,1),B(1))
  SELECT THE APPROPRIATE SEGMENT (1) OR (2) DEPENDING ON
  WHETHER ONE OR MORE MATERIALS ARE PRESENT IN THE STRUCTURE

  REWIND 19
  IF(NMAT.EQ.1) GO TO 300

  (1) THIS SEGMENT IS USED FOR THE CASE OF MORE THEN ONE
  MATERIALS PRESENT IN THE STRUCTURE

  REWIND 16
  N2=-1
  DO 200 I=1,NEL
  READ(19) COREL
  READ(16) NCON21
  N=NCON21
  IF(N.EQ.N2) GO TO 100
  CALL ELPROP(N)
  N2=N
  CALL CUR
  CALL WDISK1(I)
  CONTINUE
  RETURN
100
200

  (2) THIS SEGMENT IS USED FOR THE CASE OF ONE MATERIAL

  CALL ELPROP(1)
  DO 400 I=1,NEL
  READ(19) COREL
  CALL CUR
  CALL WDISK1(I)
  CONTINUE
  RETURN
  END
300
400

```

```

FST000020
FST000030
FST000040
FST000050
FST000060
FST000070
FST000080
FST000090
FST000100
FST000110
FST000120
FST000130
FST000140
FST000150
FST000160
FST000170
FST000180
FST000190
FST000200
FST000210
FST000220
FST000230
FST000240
FST000250
FST000260
FST000270
FST000280
FST000290
FST000300
FST000310
FST000320
FST000330
FST000340
FST000350
FST000360
FST000370
FST000380
FST000390
FST000400
FST000410
FST000420
FST000430
FST000440
FST000450
FST000460

```

SUBROUTINE CUB

LISTINGS OF SUBROUTINES PERFORMING NUMERICAL INTEGRATION BY USE OF
TWO, THREE, FOUR, OR FIVE GAUSS POINTS ARE SHOWN IN APPENDIX

SUBROUTINE FORMK(X,Y,Z,INDIC)

```
IMPLICIT REAL*8(A-H,O-Z)
COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NN,MF,NS,NCOUNT,NST,NSTF,NCLD,
1NPRC,NPSC2,NBCL,NBCH,NMAT,NCUN21,NDFP,NLINE,NREC7,KEL
COMMON /B2/ ELAST(6,6),COORD(3),COREL(2,3),UJNT(3),SSSS1(7),
1SSNN1(7)
COMMON /B3/ AK1(60,60),AK2(60,60),AK3(60,60),R01(60),R02(60),
1R03(60)
DIMENSION STK(60,60),AK(60,60),B(6,6),B1(60,6),W1(3,20),
1DNX(3,20),AJIN(3,3),AJ(3,3),CORDG(20,3)
1EQUVALENCE (STK(1,1),AK1(1,1)),(AK(1,1),AK2(1,1)),(B(1,1),
1AK2(1,1)),(B1(1,1),AK3(1,1,7)),(W1(1,1),FB2(1))
DATA CORDG/1.0D0,0.0D0,-1.0D0,-1.0D0,0.0D0,0.0D0,1.0D0,1.0D0,
11.0D0,-1.0D0,-1.0D0,1.0D0,1.0D0,0.0D0,-1.0D0,-1.0D0,0.0D0,
21.0D0,1.0D0,1.0D0,1.0D0,0.0D0,-1.0D0,-1.0D0,-1.0D0,0.0D0,
31.0D0,1.0D0,-1.0D0,-1.0D0,1.0D0,1.0D0,0.0D0,-1.0D0,-1.0D0,
4-1.0D0,0.0D0,1.0D0,1.0D0,1.0D0,1.0D0,1.0D0,1.0D0,1.0D0,
50.0D0,0.0D0,0.0D0,0.0D0,-1.0D0,-1.0D0,-1.0D0,-1.0D0,
6-1.0D0,-1.0D0/
```

GENERAL FORMULAE FOR COMPUTATION OF DERIVATIVES OF SHAPE
FUNCTIONS WITH RESPECT TO 'XI','ETA','ZETA',

FMK00020
FMK00030
FMK00040
FMK00050
FMK00060
FMK00070
FMK00080
FMK00090
FMK00100
FMK00110
FMK00120
FMK00130
FMK00140
FMK00150
FMK00160
FMK00170
FMK00180
FMK00190
FMK00200
FMK00210
FMK00220
FMK00230
FMK00240
FMK00250
FMK00260


```

CCC      (1) CORNER NODES
          DF(C,D,E,X1,Y1,Z1)=X1*(1.000+D*Y1)*(1.000+E*Z1)*(2.000*C*X1+D*Y1+
          1E*Z1-1.000)/8.000
          CCC
          (2) MIDSIDE NODES
          D2(C,D,E,      Y1,Z1)=-C*(1.000+D*Y1)*(1.000+E*Z1)/2.000
          D4(C,  E,      Y1,Z1)=(1.000-C*C)*Y1*(1.000+E*Z1)/4.000
          C*****
          COMPUTATION OF DERIVATIVES OF SHAPE FUNCTIONS WITH RESPECT
          TO 'XI','ETA','ZETA' EVALUATED AT THE POINTS OF INTEGRATION
          C*****
          (1) CORNER NODES :      XI= + 0.3 - 1
                                ZETA= + 0.3 - 1
                                ETA= + 0.3 - 1
          C*****
          NODE # : 1,3,5,7,13,15,17,19
          C*****
          DO 100 I=1,2
          II=12*(I-1)+1
          IT=II+6
          DO 100 J=11,IT,2
          XI=CORDG(J,1)
          Y1=CORDG(J,2)
          Z1=CORDG(J,3)
          W1(1,J)=DF(X,Y,Z,X1,Y1,Z1)
          W1(2,J)=DF(Y,Z,X,Y,Z1,X1,Y1)
          W1(3,J)=DF(Z,X,Y,Z1,X1,Y1)
          100
          C*****
          (2) MIDSIDE NODES :      XI= 0
                                ZETA= + 0.3 - 1
                                ETA= + 0.3 - 1
          C*****
          NODE # : 2,6,14,18
          C*****
          DO 200 K=1,2
          II=(K-1)*12+2
          IT=II+4
          DO 200 L=11,IT,4
          X1=CORDG(L,1)
          Y1=CORDG(L,2)
          Z1=CORDG(L,3)
          W1(1,L)=D2(X,Y,Z,      Y1,Z1)
          W1(2,L)=D4(X,  Z,      Y1,Z1)
          C*****
          FMK00270
          FMK00280
          FMK00290
          FMK00300
          FMK00310
          FMK00320
          FMK00330
          FMK00340
          FMK00350
          FMK00360
          FMK00370
          FMK00380
          FMK00390
          FMK00400
          FMK00410
          FMK00420
          FMK00430
          FMK00440
          FMK00450
          FMK00460
          FMK00470
          FMK00480
          FMK00490
          FMK00500
          FMK00510
          FMK00520
          FMK00530
          FMK00540
          FMK00550
          FMK00560
          FMK00570
          FMK00580
          FMK00590
          FMK00600
          FMK00610
          FMK00620
          FMK00630
          FMK00640
          FMK00650
          FMK00660
          FMK00670
          FMK00680
          FMK00690
          FMK00700
          FMK00710
          FMK00720
          FMK00730
          FMK00740

```



```

200 W1(3,L)=D4(X, Y, Z1,Y1)
CC
C(3) MIDSIDE NODES : XI= + OR - 1
ZETA= + OR - 1
ETA= 0
NODE # : 4,8,16,20
DO 300 K=1,2,4
II=(K-1)*12+4
IT=II+4
DO 300 L=11,IT,4
X1=CORRG(L,1)
Y1=CORRG(L,2)
Z1=CORRG(L,3)
W1(1,L)=D4(Y, Z,
W1(2,L)=D2(Y,Z,X,
W1(3,L)=D4(Y, X,
300
C(4) MIDSIDE NODES : XI= + OR - 1
ZETA= 0
ETA= + OR - 1
NODE # : 9,10,11,12
DO 400 L=9,12
X1=CORRG(L,1)
Y1=CORRG(L,2)
Z1=CORRG(L,3)
W1(1,L)=D4(Z, Y,
W1(2,L)=D4(Z, X,
W1(3,L)=D2(Z,X,Y,
400
C*****
C DO 500 I=1,3
DO 500 J=1,3
AJ(I,J)=0.000
C
C FORMATION OF JACOBIAN MATRIX AND COMPUTATION OF DETERMINANT
DO 500 K=1,NPEL
AJ(I,J)=AJ(I,J)+W1(I,K)*COREL(K,J)
DTJ=AJ(1,1)*AJ(2,2)*AJ(3,3)+AJ(1,2)*AJ(2,3)*AJ(3,1)-AJ(1,3)*AJ(2,1)*AJ(3,2)
2AJ(2,1)*AJ(1,2)
500
C COMPUTATION OF DERIVATIVES OF SHAPE FUNCTIONS WITH RESPECT
C

```

```

FMK00750
FMK00760
FMK00770
FMK00780
FMK00790
FMK00800
FMK00810
FMK00820
FMK00830
FMK00840
FMK00850
FMK00860
FMK00870
FMK00880
FMK00890
FMK00900
FMK00910
FMK00920
FMK00930
FMK00940
FMK00950
FMK00960
FMK00970
FMK00980
FMK00990
FMK01000
FMK01010
FMK01020
FMK01030
FMK01040
FMK01050
FMK01060
FMK01070
FMK01080
FMK01090
FMK01100
FMK01110
FMK01120
FMK01130
FMK01140
FMK01150
FMK01160
FMK01170
FMK01180
FMK01190
FMK01200
FMK01210
FMK01220

```


MULTIPLICATION OF MATRICES AND DETERMINANT OF JACOBIAN IN THE INTEGRANT OF STIFFNESS MATRIX FORMULA

```

C
DO 800 I=1,NST
DO 750 J=1,3
DO 750 K=1,3
    B1(I,J)=B1(I,J)+B(K,I)*ELAST(K,J)
750
DO 800 L=4,6
DO 800 M=1,3
    B1(I,L)=B1(I,L)+B(M,I)*ELAST(L,M)
800
DO 900 I=1,NST
DO 900 J=1,NST
    AK(I,J)=O.ODO
DO 900 K=1,6
    AK(I,J)=AK(I,J)+B1(I,K)*B(K,J)
900
DO 1000 I=1,NST
DO 1000 J=1,NST
    AK(I,J)=AK(I,J)*DTJ
1000
C
DO 1000 I=1,NST
DO 1000 J=1,NST
    AK(J,I)=AK(I,J)
1000
END

```

卷之六

```

SUBROUTINE MERGE
  IMPLICIT REAL*8 (A-H,O-Z)
  COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NN,M3,NS,NCOUNT,NST,NSTF,NCLD,
1  INPBC,NPBC2,NBCL,NBCH,NMAT,NCON21,NDFP,NLINE,NREC7,KEL
  COMMON /NB2/ NCON(20),NBC(4)
  COMMON /B3/ AK1(60,60),AK2(60,60),AK3(50,60),RB1(60),RB2(60),
1  RB3(60)
  DIMENSION A(60,60),S(60,60),B(3600),T(3600),LM(20)
  EQUIVALENCE (AK2(1,1),A(1,1),B(1),{AK1(1,1),S(1,1),T(1)},
1  (LM(1),RB1(1))

  NODE NUMBERS REFERRED TO IN THIS SUBROUTINE BELONG TO THE
  NUMBERING SYSTEM OF THE TOTAL STRUCTURE AND NOT LOCAL
  ELEMENT NUMBERING

  NTRK(I,J)=SEQUENCE NUMBER OF BLOCK POSITIONED IN ROW 'I'
  AND COLUMN 'J'

```



```

CC CC      BLOCKS ARE COUNTED ROW WISE
CC CC      NTRK(I,J)=(I-1)*(MM)+J
CC CC      ZRQ=0.000
CC CC      FILL ALL BLOCKS OF TOTAL STIFFNESS MATRIX IN ROW SEQUENCE
CC CC      BY PICKING THE PROPER ELEMENTS FROM THE INDIVIDUAL
CC CC      ELEMENT STIFFNESS MATRICES
CC CC      DO 400 I=1,NN
CC CC      DO 400 J=1,MM
CC CC      DO 100 I1=1,NS
CC CC      DO 100 J1=1,NS
CC CC      100 A(I1,J1)=ZRQ
CC CC      EXAMINE ALL ELEMENT STIFFNESS MATRICES FOR TERMS BELONGING
CC CC      TO THE BLOCK (I,J) AND TRANSFER THEM ACCORDINGLY
CC CC      REWIND 9
CC CC      DO 300 K=1,NEL
CC CC      READ(9) NCON
CC CC      DETERMINE THE LOWEST AND HIGHEST NODE NUMBER FOR EACH
CC CC      ELEMENT
CC CC      DO 200 L=1,NPEL
CC CC      LM(L)=NDF*NCON(L)-NDF
CC CC      LMIN=5000
CC CC      LMAX=0
CC CC      DO 210 I2=1,NPEL
CC CC      LMIN=MINO(LMIN,LM(I2))
CC CC      LMAX=MAXO(LMAX,LM(I2))
CC CC      LMIN=LMIN+1
CC CC      LMAX=LMAX+1
CC CC      NSI=NS*I
CC CC      NSIM=NSI-NS
CC CC      IF NO NODE OF THE ELEMENT EXAMINED BELONGS TO BLOCK (I,J)
CC CC      GO TO THE NEXT ELEMENT
CC CC      IF(LMIN.GT.NSI) GO TO 300
CC CC      IF(LMAX.LE.NSIM) GO TO 300
CC CC      IF ONE OR MORE NODES OF THE ELEMENT EXAMINED BELONG TO
CC CC      BLOCK (I,J) READ THE ELEMENT STIFFNESS MATRIX FROM DISK
CC CC      CALL RDISK1(K)

```

MRG000210
 MRG000220
 MRG000230
 MRG000240
 MRG000250
 MRG000260
 MRG000270
 MRG000280
 MRG000290
 MRG000300
 MRG000310
 MRG000320
 MRG000330
 MRG000340
 MRG000350
 MRG000360
 MRG000370
 MRG000380
 MRG000390
 MRG000400
 MRG000410
 MRG000420
 MRG000430
 MRG000440
 MRG000450
 MRG000460
 MRG000470
 MRG000480
 MRG000490
 MRG000500
 MRG000510
 MRG000520
 MRG000530
 MRG000540
 MRG000550
 MRG000560
 MRG000570
 MRG000580
 MRG000590
 MRG000600
 MRG000610
 MRG000620
 MRG000630
 MRG000640
 MRG000650
 MRG000660
 MRG000670
 MRG000680

CCCC

LOCATE THE ELEMENTS OF THE STIFFNESS MATRIX EXAMINED THAT
REFER TO BLOCK (I,J) AND TRANSFER THEM TO THE BLOCK

```

DO 290 II=1,NPEL
DO 280 JJ=1,NPEL
DO 270 KK=1,NDF
  IJ=LM(IJ)+KK-NS*(I-1)
  IF((III.GT.NS).OR.(III.LT.1)) GO TO 270
  KKK=NDF*II-NDF+KK
DO 260 LL=1,NDF
  JJ=LM(JJ)+LL-NS*(J+I-2)
  IF((JJJ.GT.NS).OR.(JJJ.LT.1)) GO TO 260
  LLL=NDF*JJ-NDF+LL
  A(III,JJJ)=A(III,JJJ)+S(KKK,LLL)
CONTINUE
260 CONTINUE
270 CONTINUE
280 CONTINUE
290 CONTINUE
300 CONTINUE

```

CCC

WRITE THE COMPLETED BLOCK ON DISK

```

NTK=NTRK(I,J)
CALL WRDISK(NTK,B)
CONTINUE

```

400

CCCC

IF THE LAST BLOCK IS NOT COMPLETELY FILED REPLACE THE
ZERO DIAGONAL ELEMENTS BY UNITS

```

IF(NN*NS.EQ.NEQ) GO TO 600
NTK=NTRK(NN,I)
CALL RDDISK(NTK,B)
DO 500 I=1,NS
  IF(DABS(A(I,I)).LT.1.0D-14) A(I,I)=1.0D0
CONTINUE
CALL WRDISK(NTK,B)

```

500

C

```

CONTINUE
RETURN
END

```

600

CCCC

MRG00690
MRG00700
MRG00710
MRG00720
MRG00730
MRG00740
MRG00750
MRG00760
MRG00770
MRG00780
MRG00790
MRG00800
MRG00810
MRG00820
MRG00830
MRG00840
MRG00850
MRG00860
MRG00870
MRG00880
MRG00890
MRG00900
MRG00910
MRG00920
MRG00930
MRG00940
MRG00950
MRG00960
MRG00970
MRG00980
MRG00990
MRG01000
MRG01010
MRG01020
MRG01030
MRG01040
MRG01050
MRG01060
MRG01070
MRG01080
MRG01090

FLD000020
FLD000030
FLD000040
FLD000050
FLD000060
FLD000070
FLD000080
FLD000090
FLD000100
FLD000110
FLD000120
FLD000130
FLD000140
FLD000150
FLD000160
FLD000170
FLD000180
FLD000190
FLD000200
FLD000210
FLD000220
FLD000230
FLD000240
FLD000250
FLD000260
FLD000270
FLD000280
FLD000290
FLD000300
FLD000310
FLD000320
FLD000330
FLD000340
FLD000350
FLD000360
FLD000370
FLD000380
FLD000390
FLD000400
FLD000410
FLD000420
FLD000430
FLD000440
FLD000450
FLD000460
FLD000470
FLD000480
FLD000490

```

C      SUBROUTINE FLOAD
C      IMPLICIT REAL*8(A-H,O-Z)
C      COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NN,MH,NS,NCOUNT,NST,NSTF,NCLD,
1  INPBC,NPBC2,NBCL,NBCH,NMAT,NCOUN21,NDFP,HL,INE,NREC7,KEL
C      COMMON /B1/ ELDAT(10,3),CLOAD(4),BCDN(3)
C      COMMON /B3/ AK1(60,60),AK2(60,60),AK3(60,60),RB1(60),RB2(60),
1  RB3(60)
C      DIMENSION A(60,60),S(60,60),R1(60),R2(60)
C      EQUIVALENCE(A(1,1),AK1(1,1)),(S(1,1),AK2(1,1)),(R1(1),RB1(1)),
1  (R2(1),RB2(1))
C      ZRO=0.000
C      REWIND 14
C      READ(14) CLOAD
C      INDEX=1
C      INCH=CLOAD(NDFP)*NDF
C      EXAMINE EACH RIGHT HAND BLOCK AND POSITION IN IT ALL
C      CORRESPONDING NODAL LOAD VECTOR:
C
100  DO 1100 I=1,NN
C      DO 100 J=1,NS
C      R1(J)=ZRO
C      IF(I*NS
50  IF(INCH.GT.IH) GO TO 1000
C      IRI=INCH-(I-1)*NS-NDF
C      DO 150 K=1,NDF
C      IR=IR1+K
150  R1(IR)=CLOAD(K)
C      IF THE LOAD POSITIONED IN THE BLOCK IS THE LAST ONE STOP
C      THE PROCESS
C      IF(INDEX.EQ.NCLD) GO TO 1200
C      READ(14) CLOAD
C      INDEX=INDEX+1
C      INCH=CLOAD(NDFP)*NDF
C      GO TO 50
C      STORE COMPLETED BLOCK ON DISK
C
1000 CALL WDSKR1(I)
1100 CONTINUE
1200 RETURN
C      CALL WDSKR1(I)
C      I1=I+1

```



```

IF(I1,GT,NN) RETURN
DO 1300 K=I1,NN
DO 1350 L=1,NS
1350 R1(L)=ZRO
CALL WDSKRI(K)
1300 CONTINUE
RETURN
END

```

SUBROUTINE BCOND
EC000020

```

IMPLICIT REAL*8(A-H,O-Z)
COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NN,M,NNS,NCOUNT,NST,NSTF,NCLD,
1NPBC,NPBC2,NBCL,NBCH,NMAT,NCON21,NDFP,MLINE,NREC7,KEL
COMMON /NB2/NCON(20),N3C(4)
COMMON /B1/ ELDAT(10,3),CLOAD(4),BCON(3)
COMMON /B3/ AK1(60,60),AK2(60,60),AK3(60,60),RB1(60),RB2(60),

```

```

1RB3(60)
DIMENSION AKW1(3600), AKW2(3600)
EQUIVALENCE (AKW1(1), AK1(1,1)), (AKW2(1,1), AK2(1,1))

```

```

IN THE COMMENT STATEMENTS THROUGHOUT THIS SUBROUTINE THE
FOLLOWING DEFINITIONS APPLY: C,P,U: THE SPECIFIED QUANTITY
READ : BEING INPUT DIRECT ACCESS STORAGE
BLOCK (I,J) : COEFFICIENT MATRIX BLOCK OF ROW 'I' AND
COLUMNS 'J'.
R.H. VECTOR : ROW AND COLUMN IDENTIFICATIONS REFER TO
THE MATRIX AS IS STORED (NN*MM). EXCEPTION
IS MADE WHEN REFERRING TO SYMMETRIC BLOCKS
R.H. VECTOR : RIGHT HAND SIDE VECTOR CORRESPONDING TO
REDUCTION : COEFFICIENT OF RIGHT HAND VECTOR AND/OR
NTRK(I,J) : COEFFICIENT BLOCK AS REQUIRED
NTRK(I,J) : SEQUENCED NUMBER OF COEFFICIENT BLOCK (I,J)
COUNTED ROW-WISE IN THE NN*MM STIFFNESS
MATRIX

```

$$\text{NTRK}(I, J) = (I - 1) * (M - 1) + J$$

EC0000290
RBC0000300
RBC0000400
RBC0000500
RBC0000600
RBC0000700
RBC0000800
RBC0000900
RBC0001000
RBC0001100
RBC0001200
RBC0001300
RBC0001400
RBC0001500
RBC0001600
RBC0001700
RBC0001800
RBC0001900
RBC0002000
RBC0002100
RBC0002200
RBC0002300
RBC0002400
RBC0002500
RBC0002600
RBC0002700
RBC0002800
RBC0002900
RBC0003000
RBC0003100
RBC0003200
RBC0003300


```

C***BC000340
C***BC000350
C***BC000360
C***BC000370
C***BC000380
C***BC000390
C***BC000400
C***BC000410
C***BC000420
C***BC000430
C***BC000440
C***BC000450
C***BC000460
C***BC000470
C***BC000480
C***BC000490
C***BC000500
C***BC000510
C***BC000520
C***BC000530
C***BC000540
C***BC000550
C***BC000560
C***BC000570
C***BC000580
C***BC000590
C***BC000600
C***BC000610
C***BC000620
C***BC000630
C***BC000640
C***BC000650
C***BC000660
C***BC000670
C***BC000680
C***BC000690
C***BC000700
C***BC000710
C***BC000720
C***BC000730
C***BC000740
C***BC000750
C***BC000760
C***BC000770
C***BC000780
C***BC000790
C***BC000800
C***BC000810

      THE FIRST PART OF SUBROUTINE 'BCND' IS USED ONLY IN THE
      SOLUTION OF PROBLEMS WHICH DO NOT INCLUDE B.C.'S WITH
      PREDEFINED DISPLACEMENTS

      ABIGN=1,OD20
      REWIND 15
      READ(15) NBC
      NBC1=NBC(1)+NDF
      INDEX=1

      ALL DIAGONAL BLOCKS ARE EXAMINED AND REDUCED IF REQUIRED
      IN SEQUENCE

      DO 150 I=1,NN
      IH=I*NS
      IL=(I-1)*NS+1

      IF THE B.C. TO BE APPLIED DOES NOT CORRESPOND TO THE
      DIAGONAL BLOCK (I,I) EXAMINE THE NEXT DIAGONAL BLOCK

100  IF(NBC1.GT.IH) GO TO 150
      NTK=NTRK(I,1)
      CALL RDDISK(NTK,AKW1)
110  IBL=NBC(1)+1-NS*(I-1)
      IBH=IBL+NDF-1
      IC=1
      DO 120 K=IBL,IBH
      IC=IC+1

      IF THE B.C. CODE CORRESPONDING TO DIAGONAL ELEMENT (K,K)
      IS NOT 'I' NO MODIFICATION IS REQUIRED

      IF(NBC(IC).NE.1) GO TO 120
      AK1(K,K)=AK1(K,K)+ABIGN
      CONTINUE

120  IF THE B.C. APPLIED IS THE LAST OF THE PROBLEM NO FURTHER
      MODIFICATION OF COEFFICIENT MATRIX IS REQUIRED

      IF(INDEX.EQ.NPBC) GO TO 130
      READ(15) NBC
      NBC1=NBC(1)+NDF

```



```

C*** DO 900 I=IR1,NN
C   IFR1=IFR-I+1
C   IF(IFR1.GT.MM) GO TO 900
C   REWIND 15
C   REWIND 22
C
C   READ THE RIGHT HAND VECTOR CORRESPONDING TO ROW 'I'
C
C   NTKR1=I
C   CALL RDSKR1(NTKR1)
C
C   READ THE FIRST NOT APPLIED B.C.
C
C   DO 200 L=1,I1
C     READ(15) NBC
C     NBC1=NBC(1)+NDF
C     NBC2=NBC1/NDF
C
C   SET THE POINTER ON FILE 22 AT THE FIRST NOT YET APPLIED
C   PREDEFINED DISPLACEMENT
C
C   IF(I5.EQ.0) GO TO 203
C   DO 202 L=1,I5
C     READ(22) BCON
C
C   DETERMINATION OF FIRST BLOCK OF ROW 'I' TO BE REDUCED AND
C   TRANSFER OF CONTROL TO THE APPROPRIATE SEGMENT
C
C   SELECT THE PROPER SEGMENT DEPENDING ON WHETHER THE FIRST
C   BLOCK TO BE REDUCED IS DIAGONAL OR NOT
C
C   IFR1=IFR
C   J3=2
C   IF(IFR.NE.I) GO TO 500
C
C   REDUCTION OF DIAGONAL BLOCK AND RIGHT HAND BLOCK OF ROW 'I'
C
C   I4=0
C   IH=I*NS

```



```

C      READ THE 'I' DIAGONAL BLOCK
C      NTK1=NTRK(I,I)
C      CALL RDDISK(NTK1,AKW1)
C      EXAMINE THE B.C. CODES FOR NODE NBC2 AND PERFORM THE
C      NECESSARY MODIFICATIONS TO THE DIAGONAL BLOCK AND THE
C      RIGHT HAND VECTOR
C      I61=0
C      I6=0
C      DO 205 K=1,NDF
C      K1=K+1
C      M=NBC(K1)+1
C      I1=NBC(I1)-(I-1)*NS+K
C      GO TO (250,230,220),M
C      IF (I6.NE.0) GO TO 222
C      READ(22) BCON
C      I5=I5+1
C      I6=I
C      I61=1
C      DO 225 L=1,NS
C      RB1(L)=RB1(L)-AK1(L,I1)*BCON(K)
C      I4=I4+1
C      BCON1(I4,1)=I1
C      BCON1(I4,2)=BCON(K)
C      AK1(I1,I1)=AK1(I1,I1)+ABIGN
C      CONTINUE
C      J3=MM+1
C      MML=I/MMF-(I-MMF)/MMF
C      MM1=MM-(I-MMF)*MML
C      J4=MM1
C      IF (NBC2.EQ.NBCH) GO TO 410
C      IF THE B.C. APPLIED IS THE LAST ONE, NO FURTHER REDUCTION
C      OF ROW 'I' IS REQUIRED
C      PROCEED WITH REDUCTION OF BLOCKS SYMMETRIC TO THE ONES OF
C      ROW 'I'
C      READ THE NEXT B.C.
C      READ(15) NBC
C      NBC1=NBC(I1)+NDF
C      NBC2=NBC1/NDF
C      I1=I1+1
C      IF THE NEW B.C. DOES NOT CORRESPOND TO THE DIAGONAL BLOCK

```



```

CC002260
CC002270
CC002280
CC002290
CC002300
CC002310
CC002320
CC002330
CC002340
CC002350
CC002360
CC002370
CC002380
CC002390
CC002400
CC002410
CC002420
CC002430
CC002440
CC002450
CC002460
CC002470
CC002480
CC002490
CC002500
CC002510
CC002520
CC002530
CC002540
CC002550
CC002560
CC002570
CC002580
CC002590
CC002600
CC002610
CC002620
CC002630
CC002640
CC002650
CC002660
CC002670
CC002680
CC002690
CC002700
CC002710
CC002720
CC002730

PROCEED WITH REDUCTION OF NON DIAGONAL BLOCKS AND THEIR
SYMMETRICS

IF(NBC1.LE.IH) GO TO 205

*****
AFTER COMPLETION OF REDUCTION OF DIAGONAL BLOCK THE
FOLLOWING STEPS ARE EXECUTED
(1) DETERMINATION OF BLOCK 'J1' CORRESPONDING TO THE
LAST B.C. READ FROM DATA
(2) IF BLOCK 'J1' IS THE NEXT TO THE DIAGONAL, CONTROL
IS TRANSFERRED TO THE SEGMENT REDUCING THE RIGHT HAND
VECTOR CORRESPONDING TO NON DIAGONAL BLOCKS AND
THEIR SYMMETRICS
(3) IF STEP (2) IS NOT EXECUTED REDUCE THE RIGHT HAND
VECTORS CORRESPONDING TO BLOCKS SYMMETRIC TO THE
ONES BETWEEN THE DIAGONAL AND 'J1'.
*****

400 STEP (1)
J1=(NBC1+NS-1)/NS
J2=J1+1
J3=J1-I+1
IFR=J1
IH=J1*NS

410 IF(J1.EQ.J2) GO TO 490
STEP (2)

430 IF(J1.EQ.J2) GO TO 490
STEP (3)

J4=J3-1
IF(J4.GT.MM) J4=MM
IF(I.EQ.NN) GO TO 950
DO 450 J=2,J4
NTK2=NTKR(I,J)
CALL RDDISK(NTK2,AKW2)
NTKR2=J+I-1
CALL RDSKR2(NTKR2)
DO 430 L1=1,I4
IR=BCON1(L1,I)
DO 430 L2=1,NS
RB2(L2)=RB2(L2)-AK2(IR,L2)*BCON1(L1,2)

```



```

C 450 CALL WDSKR2(NTKR2)
C *****
C 490 CALL WRDISK(NTK1,AKW1)
500 IF(J3.GT.MM) GO TO 850
    MML=I/MMF-(I-MMF)/MMF
    MML=MM-(I-MMF)*MML
    DO 800 J=J3,MM1
    IH=(J+(I-1))*NS
        READ BLOCK (I,J)
    NTK2=NTKR(I,J)
    CALL RDISK(NTK2,AKW2)
    IHL=(J+I-2)*NS+1
    IPD=(IH+1-NBC1)*((IHL-NBC1)
        IF THE B.C. NBC2 DOES NOT CORRESPOND TO BLOCK (I,J)
        PROCEED WITH REDUCTION OF ITS SYMMETRIC BLOCK
    IF(IPD.GT.0) GO TO 600
        MODIFY RIGHT HAND VECTOR 'I' FOR ALL B.C.'S CORRESPONDING
        TO BLOCK (I,J)
510 I6=0
    DO 560 K=1,NDF
        K1=K+1
        IF(NBC(K1).NE.2) GO TO 560
        IF(I6.NE.0) GO TO 550
        READ(22) BCON
        I6=1
550 II=NBC(1)+K-(I+J-2)*NS
    DO 555 L=1,NS
555 RB1(L)=RB1(L)-AK2(L,II)*BCON(K)
560 CONTINUE
        IF THE B.C. APPLIED IS THE LAST ONE, NO FURTHER REDUCTION
        IN ROW 'I' IS REQUIRED
        PROCEED WITH REDUCTION OF ITS SYMMETRIC BLOCK
    IF(NBC2.EQ.NBCH) GO TO 600
    READ(15) NBC
    NBC1=NBC(1)+NDF
    NBC2=NBC1/NDF
        IF THE NEW B.C. DOES NOT CORRESPOND TO BLOCK (I,J) PROCEED

```

BC002740
 BC002750
 BC002760
 BC002770
 BC002780
 BC002790
 BC002800
 BC002810
 BC002820
 BC002830
 BC002840
 BC002850
 BC002860
 BC002870
 BC002880
 BC002890
 BC002900
 BC002910
 BC002920
 BC002930
 BC002940
 BC002950
 BC002960
 BC002970
 BC002980
 BC002990
 BC003000
 BC003010
 BC003020
 BC003030
 BC003040
 BC003050
 BC003060
 BC003070
 BC003080
 BC003090
 BC003100
 BC003110
 BC003120
 BC003130
 BC003140
 BC003150
 BC003160
 BC003170
 BC003180
 BC003190
 BC003200
 BC003210


```

CALL SYMINV(AK2,NS,RB1,RB2,IPLG)
      (1B) IF A(N,1) IS SINGULAR GIVE ERROR MESSAGE AND
            STOP EXECUTION FOR THIS PROBLEM
            IF A(N,1) IS NEARLY SINGULAR GIVE WARNING
            MESSAGE AND CONTINUE EXECUTION
IF(IPLG.EQ.1) CALL ERROR4(N)
IF(IPLG.EQ.2) CALL ERROR5(N)
      (1C) READ B(N)
            MULTIPLY B(N) BY A*(N,1)
            STORE THE PRODUCT IN PLACE OF B(N)
NTR=N
CALL RDSKR1(NTR)
CALL MULT(AK2,RB1,RB2,NS,NS,1)
CALL WDSKR2(NTR)
      (1D) IF ROW 'N' IS THE LAST ROW SKIP THE FOLLOWING
            STEPS AND START BACK SUBSTITUTION
IF(N.EQ.NN) GO TO 300
      (2) REDUCTION OF NON DIAGONAL COEFFICIENT BLOCKS OF
            ROW 'N'
            IF BLANK BLOCKS EXIST SKIP THEM
IF(N.GT.(NN-MM+1)) KMM=KMM-1
            THE FOLLOWING STEPS ARE REPEATED FOR ALL BLOCKS A(N,K)
      (2A) READ A(N,K)
            MULTIPLY A(N,K) BY A*(N,1)
            STORE THE PRODUCT IN THE PLACE OF A(N,K)
DO 200 K=2,KMM
NTR=NTRK(N,K)
CALL RDDISK(NTR,AK1)
CALL MULT(AK2,AK1,AK3,NS,NS,NS)
CALL WRDISK(NTR,AK3)

```

SLV00530
 SLV00590
 SLV00600
 SLV00610
 SLV00620
 SLV00630
 SLV00640
 SLV00650
 SLV00660
 SLV00670
 SLV00680
 SLV00690
 SLV00700
 SLV00710
 SLV00720
 SLV00730
 SLV00740
 SLV00750
 SLV00760
 SLV00770
 SLV00780
 SLV00790
 SLV00800
 SLV00810
 SLV00820
 SLV00830
 SLV00840
 SLV00850
 SLV00860
 SLV00870
 SLV00880
 SLV00890
 SLV00900
 SLV00910
 SLV00920
 SLV00930
 SLV00940
 SLV00950
 SLV00960
 SLV00970
 SLV00980
 SLV00990
 SLV01000
 SLV01010
 SLV01020
 SLV01030
 SLV01040
 SLV01050

00000000 00000000 0000 000000 000000000000

SLV01060
SLV01070
SLV01080
SLV01090
SLV01100
SLV01110
SLV01120
SLV01130
SLV01140
SLV01150
SLV01160
SLV01170
SLV01180
SLV01190
SLV01200
SLV01210
SLV01220
SLV01230
SLV01240
SLV01250
SLV01260
SLV01270
SLV01280
SLV01290
SLV01300
SLV01310
SLV01320
SLV01330
SLV01340
SLV01350
SLV01360
SLV01370
SLV01380
SLV01390
SLV01400
SLV01410
SLV01420
SLV01430
SLV01440
SLV01450
SLV01460
SLV01470
SLV01480
SLV01490
SLV01500
SLV01510
SLV01520
SLV01530

(2B) COMPUTE A'(N,K)

STORE A'NN,K) IN THE BLANK BLOCK A(NN,K)

DO 150 I=1,NS
II=(I-1)*NS
DO 150 J=1,NS
IL=II+J
IR=I+(J-1)*NS
AK3(IL)=AK1(IR)
NTK=NTRK(NN,K)
CALL WRDISK(NTK,AK3)
CONTINUE

150

200

(3) REDUCE THE REMAINING ROWS STARTING WITH ROW 'N+1'
THE FOLLOWING STEPS ARE REPEATED FOR EVERY ROW 'I'

DO 260 L=2,KMM
I=N+L-1
IF(I.GT.NN) GO TO 260
J=0

(2A) READ A'(N,L) STORED IN BLOCK (NN,L)

NTK=NTRK(NN,L)
CALL RDDISK(NTK,AK2)

(2B) THE FOLLOWING STEPS ARE REPEATED FOR ALL BLOCKS
OF ROW 'I'

READ A(N,K)

MULTIPLY A(N,K) BY A'(N,L)

READ A(I,J) AND SUBTRACT FROM IT THE PRODUCT
OF THE PREVIOUS STEP

STORE THE RESULT IN THE PLACE OF A(I,J)

DO 250 K=L,KMM
J=J+1

NTK=NTRK(N,K)
CALL RDDISK(NTK,AK1)
CALL MULT(AK2,AK1,AK3,NS,NS,NS)


```

NTK=NTRK(I,J)
CALL RDDISK(NTK,AK1)
DO 210 I=1,NCOUNT
210 AK1(I1)=AK1(I1)-AK3(I1)
250 CALL WRDISK(NTK,AK1)
CONTINUE

      (2C) MULTIPLY B(N) BY A'(N,L)
      READ B(I)
      FORM B(I)-B(N)*A'(N,L)
      STORE THE PRODUCT IN THE PLACE OF B(I)

      CALL MULT(AK2,RB2,RB3,NS,NS,1)
      NTR=I
      CALL RDSKR1(NTR)
      DO 255 I=1,NS
255 RB1(I1)=RB1(I1)-RB3(I1)
      CALL WDSKR1(NTR)
260 CONTINUE
      GO TO 100

*****
      BACK SUBSTITUTION
      THE FOLLOWING STEPS ARE REPEATED FOR ALL ROWS STARTING
      WITH ROW 'NN-1'

300 N=N-1
      IF THE LAST ROW REDUCED WAS ROW NO 1 RETURN

      IF(N.EQ.0) GO TO 500
      READ B(N)
      NT1=N
      CALL RDSKR3(NT1)

      THE FOLLOWING ARE REPEATED FOR ALL BLOCKS A(N,K)
      (1) READ A(N,K)
      (2) READ B(L) WHERE B(L)= VECTOR OF SOLUTIONS FOR THE
      UNKNOWNNS CORRESPONDING TO BLOCK A(N,K) FOUND IN

```

SLV01540
 SLV01550
 SLV01560
 SLV01570
 SLV01580
 SLV01590
 SLV01600
 SLV01610
 SLV01620
 SLV01630
 SLV01640
 SLV01650
 SLV01660
 SLV01670
 SLV01680
 SLV01690
 SLV01700
 SLV01710
 SLV01720
 SLV01730
 SLV01740
 SLV01750
 SLV01760
 SLV01770
 SLV01780
 SLV01790
 SLV01800
 SLV01810
 SLV01820
 SLV01830
 SLV01840
 SLV01850
 SLV01860
 SLV01870
 SLV01880
 SLV01890
 SLV01900
 SLV01910
 SLV01920
 SLV01930
 SLV01940
 SLV01950
 SLV01960
 SLV01970
 SLV01980
 SLV01990
 SLV02000
 SLV02010

SLV02020
SLV02030
SLV02040
SLV02050
SLV02060
SLV02070
SLV02080
SLV02090
SLV02100
SLV02110
SLV02120
SLV02130
SLV02140
SLV02150
SLV02160
SLV02170
SLV02180
SLV02190
SLV02200
SLV02210
SLV02220
SLV02230
SLV02240
SLV02250
SLV02260
SLV02270
DSP00010

PREVIOUS STEPS OF BACK SUBSTITUTION
(3) FORM $B(NE-A(N,K)*B(L))$
AT THE END OF THE OPERATIONS FOR ALL BLOCKS OF ROW 'N',
THE VECTOR B(N) CONTAINS THE SOLUTIONS FOR THE UNKNOWNNS
CORRESPONDING TO THE DIAGONAL BLOCK A(N,1) AND IS STORED
ON DISK

DO 400 K=2,KMM
L=N+K-1
IF(L.GT.NN) GO TO 400
NTR=NTRK(N,K)
CALL RDDISK(NTR,AK1)
NTR=L
CALL RDSKR1(NTR)
CALL MULT(AK1,RB1,RB2,NS,NS,1)
DO 310 K1=1,NS
RB3(K1)=RB3(K1)-RB2(K1)
CONTINUE
CALL WDSKR3(NT1)
KMM=KMM+1
IF(KMM.GT.MM) KMM=MM
GO TO 300
RETURN
END

310
400

500

C

SUBROUTINE DISP

IMPLICIT REAL*8(A-H,O-Z)
COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NN,MM,NS,NCOUNT,NST,NSTF,NCLD,
1NPBC,NPBC2,NBCL,NBCH,NMAT,NCON21,NDFP,KLINE,NREC7,KEL
COMMON /NB2/ NCON(20),NRC(4)
COMMON /B3/ AK1(60,60),AK2(60,60),RB1(60),RB2(60),
1RB3(60)
DIMENSION REACT(4),REACT1(3)
EQUIVALENCE (REACT(1),RB2(1)),(REACT(2),REACT1(1))

ABIGN=1.0D20
ZRO=0.0D0
REWIND 15
REWIND 17
READ(15) NRC
NRC1=NRC(1)+NDF
INDEX=1
NPBL=NS/NDF

C

DSP00020
DSP00030
DSP00040
DSP00050
DSP00060
DSP00070
DSP00080
DSP00090
DSP00100
DSP00110
DSP00120
DSP00130
DSP00140
DSP00150
DSP00160
DSP00170
DSP00180
DSP00190
DSP00200


```

C C C C C
JLL=NDPT-(NDPT/NPBL)*NPBL
IF(JLL.EQ.0) JLL=NPBL
NLI=0
WRITE(6,1000)
C C C C C
TRANSFER OF RIGHT HAND BLOCKS IN SEQUENCE FROM DISK INTO
CORE
DO 280 I=1,NN
CALL RDSKRI(I)
CALL WDISKB(I)
C C C C C
IF THE CONSTRAINED POINT EXAMINED IS NOT INCLUDED IN THE
RIGHT HAND BLOCK 'I', SKIP NEXT SEGMENT
IM1=I-1
ICTI=IM1*NPBL
IH=I*NS
IL=IM1*NS+1
ICHECK=(NBC1-IH)*(IL-NBC1)
IF(ICHECK.LT.0) GO TO 255
210
C C C C C
DETERMINE THE REACTIONS AT THE CONSTRAINED POINTS AND
MODIFY CORRESPONDINGLY THE RIGHT HAND VECTOR
DO 215 J=1,NDFP
REACT(J)=ZRO
I1=NBC(1)-NS*(I-1)+1
I2=I1+1
I3=I2+1
C
IF(NBC(2).EQ.0) GO TO 220
REACT(2)=RBI(I1)*ABIGN
RBI(I1)=ZRO
IF(NBC(3).EQ.0) GO TO 240
REACT(3)=RBI(I2)*ABIGN
RBI(I2)=ZRO
IF(NBC(4).EQ.0) GO TO 250
REACT(4)=RBI(I3)*ABIGN
RBI(I3)=ZRO
ANBC=NBC(1)/NDF+1
REACT(1)=ANBC
WRITE(17) REACT
IF(INDEX.EQ.NPBC) GO TO 255
READ(15) NBC
NBC1=NBC(1)+NDF
INDEX=INDEX+1
250
220
240
250

```

DSP00210
 DSP00220
 DSP00230
 DSP00240
 DSP00250
 DSP00260
 DSP00270
 DSP00280
 DSP00290
 DSP00300
 DSP00310
 DSP00320
 DSP00330
 DSP00340
 DSP00350
 DSP00360
 DSP00370
 DSP00380
 DSP00390
 DSP00400
 DSP00410
 DSP00420
 DSP00430
 DSP00440
 DSP00450
 DSP00460
 DSP00470
 DSP00480
 DSP00490
 DSP00500
 DSP00510
 DSP00520
 DSP00530
 DSP00540
 DSP00550
 DSP00560
 DSP00570
 DSP00580
 DSP00590
 DSP00600
 DSP00610
 DSP00620
 DSP00630
 DSP00640
 DSP00650
 DSP00660
 DSP00670
 DSP00680

DSP00690
 DSP00700
 DSP00710
 DSP00720
 DSP00730
 DSP00740
 DSP00750
 DSP00760
 DSP00770
 DSP00780
 DSP00790
 DSP00800
 DSP00810
 DSP00820
 DSP00830
 DSP00840
 DSP00850
 DSP00860
 DSP00870
 DSP00880
 DSP00890
 DSP00900
 DSP00910
 DSP00920
 DSP00930
 DSP00940
 DSP00950
 DSP00960
 DSP00970
 DSP00980
 DSP00990
 DSP01000
 DSP01010
 DSP01020
 DSP01030
 DSP01040
 DSP01050
 DSP01060
 DSP01070
 DSP01080
 DSP01090
 DSP01100
 DSP01110
 DSP01120

```

C 255 GO TO 210
      NL2=NL1+NPBL
      JL=NPBL
      IF(I.EQ.NN) JL=JLL
      IF(NL2.GT.NLINE) GO TO 262
      DO 260 M=1,JL
      ICT=ICTI+M
      K1=(M-1)*NDF+1
      K2=K1+2
      WRITE(6,2000) ICT,(RB1(N1),N1=K1,K2)
      NL1=NL2
      GO TO 280
262 J2=NLINE-NL1
      DO 264 N=1,J2
      ICT=ICTI+N
      K1=(N-1)*NDF+1
      K2=K1+2
      WRITE(6,2000) ICT,(RB1(N2),N2=K1,K2)
      NL1=NPBL-J2
      WRITE(6,1000)
      J1=J2+1
      DO 266 M1=J1,JL
      ICT=ICTI+M1
      K1=(M1-1)*NDF+1
      K2=K1+2
      WRITE(6,2000) ICT,(RB1(N3),N3=K1,K2)
266 CONTINUE
280
C 285 SUMX=ZRO
      SUMY=ZRO
      SUMZ=ZRO
      I2=ZRO
      REWIND 17
      PRINT THE REACTIONS
C
C
290 I1=1
      I2=I2+NLINE
      IF(NPBC.LE.I2) I2=NPBC
      WRITE(6,1100)
      DO 300 L3=I1,I2
      READ(17) REACT
      ICT=REACT(1)
  
```



```

C      6-1.0D0,-1.0D0/
      READ(9) NCON
      IF(KEL.EQ.0) GO TO 200
      WRITE(6,1000) I
C      FORM THE VECTOR OF ELEMENT NODAL DISPLACEMENTS
C      200 DO 300 J=1,NPEL
C          LM(J)=NCON(J)
C          JJ=LM(J)
C          JEL=3*(J-1)
C          DO 250 K=1,NDF
C              CALL RDISKB(JJ)
C              J2=JEL+K
C              UEL(J2)=UJNT(K)
C          CONTINUE
C      250
C      300
C      COMPUTATION OF ELEMENT NODAL STRAINS
C      DO 400 J=1,NPEL
C          JJ=NCON(J)
C          X=CORDG(J,1)
C          Y=CORDG(J,2)
C          Z=CORDG(J,3)
C          CALL FORMK(X,Y,Z,2)
C          DO 310 J1=1,6
C              SSSS(J1)=0.0D0
C              SSNN(J1)=0.0D0
C          DO 310 J2=1,NST
C              SSNN(J1)=SSNN(J1)+B(J1,J2)*UEL(J2)
C      310
C      COMPUTATION OF NODAL STRESSES FROM NODAL STRAINS
C      DO 320 J1=1,3
C          DO 320 J2=1,3
C              SSSS(J1)=SSSS(J1)+SSNN(J2)*ELAST(J1,J2)
C      320
C      330 DO 320 K1=4,6
C          SSSS(K1)=SSSS(K1)+SSNN(K1)*ELAST(K1,K1)
C      FORMATION OF NODAL STRESS AND STRAIN VECTORS FOR THIS
C      ELEMENT
C      CALL RDISKS(JJ)
C      CALL RDISKN(JJ)
C      SSSS1(7)=SSSS1(7)+1.0D0
C      SSNN1(7)=SSNN1(7)+1.0D0
C      DO 350 K=1,6

```



```

C      EQUIVALENCE (STK1(1),AK1(1,1)),(STK2(1),AK1(1,16)),
C      1(STK3(1),AK1(1,31)),(STK4(1),AK1(1,46))
C
C      NCOUNT=NS*NS
C      LREC7=NCOUNT/NREC7
C      NS1=NS/NDF
C      RETURN
C
C      *****
C      ENTRY WRDISK(NTRACK,A)
C
C      I7=(NTRACK-1)*NREC7+1
C      N1=1
C      DO 10 L=1,NREC7
C      N2=L*LREC7
C      WRITE(7,I7)
C      10 N1=N2+1
C      RETURN
C
C      *****
C      ENTRY RDDISK(NTRACK,A)
C
C      I7=(NTRACK-1)*NREC7+1
C      N1=1
C      DO 20 L=1,NREC7
C      N2=L*LREC7
C      READ(7,I7,ERR=500)
C      N1=N2+1
C      20 RETURN
C
C      *****
C      ENTRY WDISK1(NTRACK)
C
C      I8=(NTRACK-1)*4+1
C      WRITE(8,I8) STK1
C      WRITE(8,I8) STK2
C      WRITE(8,I8) STK3
C      WRITE(8,I8) STK4
C      RETURN
C
C      *****
C      ENTRY RDISK1(NTRACK,A)
C
C      *****

```



```
C  
C  
C      ISTOP=0  
      ISTOP9=0  
      ISTP16=0  
  
          FILE NO 7 (EXCEPT NR7)  
  
        IFILE=7  
        IF(LRB7-NCOUNT*8/NREC7) 40,60,50  
40    WRITE(6,2000)       IFILE  
      WRITE(6,8000)       IFILE  
      ISTOP=1  
      GO TO 60  
50    WRITE(6,3000)       IFILE  
      WRITE(6,8500)       IFILE  
  
60    IF(LRW7-NCOUNT*2/NREC7) 70,90,80  
70    WRITE(6,2000)       IFILE  
      WRITE(6,9000)       IFILE  
      ISTOP=1  
      GO TO 90  
80    WRITE(6,3000)       IFILE  
      WRITE(6,9500)       IFILE  
  
90    IF((LRW7*4-LRB7).EQ.0) GO TO 95  
      WRITE(6,2000)       IFILE  
      WRITE(6,9700)       IFILE  
      ISTOP=1  
  
          FILE NO 8  
  
        IFILE=8  
        IF(NR8-NEL*4) 100,120,110  
95    WRITE(6,2000)       IFILE  
100   WRITE(6,6000)       IFILE  
      ISTOP=1  
      GO TO 120  
110   WRITE(6,3000)       IFILE  
      WRITE(6,5000)       IFILE  
  
120   IF(LRB8-7200) 130,150,140  
130   WRITE(6,2000)       IFILE  
      WRITE(6,8000)       IFILE  
      ISTOP=1  
      GO TO 150
```


ERR00700
ERR00710
ERR00720
ERR00730
ERR00740
ERR00750
ERR00760
ERR00770
ERR00780
ERR00790
ERR00800
ERR00810
ERR00820
ERR00830
ERR00840
ERR00850
ERR00860
ERR00870
ERR00880
ERR00890
ERR00900
ERR00910
ERR00920
ERR00930
ERR00940
ERR00950
ERR00960
ERR00970
ERR00980
ERR00990
ERR01000
ERR01010
ERR01020
ERR01030
ERR01040
ERR01050
ERR01060
ERR01070
ERR01080
ERR01090
ERR01100
ERR01110
ERR01120
ERR01130
ERR01140
ERR01150
ERR01160
ERR01170

```

140 WRITE(6,3000)      IFILE
    C  WRITE(6,8500)
150 IF(LRW8-1800) 160,180,170
160 WRITE(6,2000)      IFILE
    WRITE(6,9000)
    ISTOP=1
    GO TO 180
170 WRITE(6,3000)      IFILE
    WRITE(6,9500)
    C  IF((LRW8*4-LRB8).EQ.0) GO TO 185
180 WRITE(6,2000)      IFILE
    WRITE(6,9700)
    ISTOP=1
    C  FILE NO 9
    C  185 IFILE=9
    IF(NR9-NEL) 190,210,200
190 WRITE(6,2000)      IFILE
    WRITE(6,6000)
    ISTOP=1
    GO TO 210
200 WRITE(6,3000)      IFILE
    WRITE(6,5000)
    C  210 IF(LRB9-80) 220,240,230
220 WRITE(6,2000)      IFILE
    WRITE(6,8000)
    ISTOP=1
    GO TO 240
230 WRITE(6,3000)      IFILE
    WRITE(6,8500)
    C  FILE NO 10
    C  240 IFILE=10
    IF(NR10-NDPT) 250,270,260
250 WRITE(6,2000)      IFILE
    WRITE(6,6000)
    ISTOP=1
    GO TO 270
260 WRITE(6,3000)      IFILE
    WRITE(6,5000)
    C  IF(LRB10-24) 280,300,290
270 IF(LRB10-24) 280,300,290
280 WRITE(6,2000)

```


ERROR1180
 ERROR1190
 ERROR1200
 ERROR1210
 ERROR1220
 ERROR1230
 ERROR1240
 ERROR1250
 ERROR1260
 ERROR1270
 ERROR1280
 ERROR1290
 ERROR1300
 ERROR1310
 ERROR1320
 ERROR1330
 ERROR1340
 ERROR1350
 ERROR1360
 ERROR1370
 ERROR1380
 ERROR1390
 ERROR1400
 ERROR1410
 ERROR1420
 ERROR1430
 ERROR1440
 ERROR1450
 ERROR1460
 ERROR1470
 ERROR1480
 ERROR1490
 ERROR1500
 ERROR1510
 ERROR1520
 ERROR1530
 ERROR1540
 ERROR1550
 ERROR1560
 ERROR1570
 ERROR1580
 ERROR1590
 ERROR1600
 ERROR1610
 ERROR1620
 ERROR1630
 ERROR1640
 ERROR1650

```

WRITE(6,8000) IFILE
I$TOP=1
GO TO 300
290 WRITE(6,3000)
WRITE(6,8500) IFILE

C 300 IF(LRW10-6) 310,330,320
310 WRITE(6,2000)
WRITE(6,9000) IFILE
I$TOP=1
GO TO 330
320 WRITE(6,3000)
WRITE(6,9500) IFILE

C 330 IF((LRW10*4-LRB10).EQ.0) GO TO 335
WRITE(6,2000)
WRITE(6,9700) IFILE
I$TOP=1

C
C FILE NO 12
C
335 IFILE=12
IF(NR12-NDPT) 340,360,350
340 WRITE(6,2000)
WRITE(6,6000) IFILE
I$TOP=1
GO TO 360
350 WRITE(6,3000)
WRITE(6,5000) IFILE

C 360 IF(LPB12-56) 370,390,380
370 WRITE(6,2000)
WRITE(6,8000) IFILE
I$TOP=1
GO TO 390
380 WRITE(6,3000)
WRITE(6,8500) IFILE

C 390 IF(LRW12-14) 400,420,410
400 WRITE(6,2000)
WRITE(6,9000) IFILE
I$TOP=1
GO TO 420
410 WRITE(6,3000)
WRITE(6,9500) IFILE

C 420 IF((LRW12*4-LRB12).EQ.0) GO TO 425
WRITE(6,2000)

```


ERR01660
ERR01670
ERR01680
ERR01690
ERR01700
ERR01710
ERR01720
ERR01730
ERR01740
ERR01750
ERR01760
ERR01770
ERR01780
ERR01790
ERR01800
ERR01810
ERR01820
ERR01830
ERR01840
ERR01850
ERR01860
ERR01870
ERR01880
ERR01890
ERR01900
ERR01910
ERR01920
ERR01930
ERR01940
ERR01950
ERR01960
ERR01970
ERR01980
ERR01990
ERR02000
ERR02010
ERR02020
ERR02030
ERR02040
ERR02050
ERR02060
ERR02070
ERR02080
ERR02090
ERR02100
ERR02110
ERR02120
ERR02130

```

C      WRITE(6,9700) IFILE
C      ISTOP=1
C      FILE NO 13
425  IFILE=13
      IF(NR13-NDPT) 430,450,440
430  WRITE(6,2000) IFILE
      ISTOP=1
      GO TO 450
440  WRITE(6,3000)
      WRITE(6,5000) IFILE
C
450  IF(LRB13-56) 460,480,470
460  WRITE(6,2000)
      WRITE(6,8000) IFILE
      ISTOP=1
      GO TO 480
470  WRITE(6,3000)
      WRITE(6,8500) IFILE
C
480  IF(LRW13-14) 490,510,500
490  WRITE(6,2000)
      WRITE(6,9000) IFILE
      ISTOP=1
      GO TO 510
500  WRITE(6,3000)
      WRITE(6,9500) IFILE
C
510  IF((LRW13*4-LRB13).EQ.0) GO TO 515
      WRITE(6,2000)
      WRITE(6,9700) IFILE
      ISTOP=1
C      FILE NO 14
C
515  IFILE=14
      IF(NR14-NCLD) 520,540,530
520  WRITE(6,2000)
      WRITE(6,6000) IFILE
      ISTOP=1
      GO TO 540
530  WRITE(6,3000)
      WRITE(6,5000) IFILE
C
540  IF(LRB14-32) 550,570,560
550  WRITE(6,2000)

```



```

WRITE(6,8000) IFILE
ISTOP=1
GO TO 570
560 WRITE(6,3000)
WRITE(6,8500) IFILE
C
C
C
FILE NO 15
570 IFILE=15
IF(NR15-NPBC) 580,600,590
580 WRITE(6,2000)
WRITE(6,6000) IFILE
ISTOP=1
GO TO 600
590 WRITE(6,3000)
WRITE(6,5000) IFILE
C
600 IF(LRB15-16) 610,630,620
610 WRITE(6,2000)
WRITE(6,8000) IFILE
ISTOP=1
GO TO 630
620 WRITE(6,3000)
WRITE(6,8500) IFILE
C
C
C
FILE NO 16
630 IF(NMAT.EQ.1) GO TO 690
IFILE=16
IF(NR16-NEL) 640,660,650
640 WRITE(6,2000)
WRITE(6,6000) IFILE
ISTOP=1
GO TO 660
650 WRITE(6,3000)
WRITE(6,5000) IFILE
C
660 IF(LRB16-4) 670,690,680
670 WRITE(6,2000)
WRITE(6,8000) IFILE
ISTOP=1
GO TO 690
680 WRITE(6,3000)
WRITE(6,8500) IFILE
C
C
C
FILE NO 17
690 IFILE=17

```

```

ERR02140
ERR02150
ERR02160
ERR02170
ERR02180
ERR02190
ERR02200
ERR02210
ERR02220
ERR02230
ERR02240
ERR02250
ERR02260
ERR02270
ERR02280
ERR02290
ERR02300
ERR02310
ERR02320
ERR02330
ERR02340
ERR02350
ERR02360
ERR02370
ERR02380
ERR02390
ERR02400
ERR02410
ERR02420
ERR02430
ERR02440
ERR02450
ERR02460
ERR02470
ERR02480
ERR02490
ERR02500
ERR02510
ERR02520
ERR02530
ERR02540
ERR02550
ERR02560
ERR02570
ERR02580
ERR02590
ERR02600
ERR02610

```


ERR02620
ERR02630
ERR02640
ERR02650
ERR02660
ERR02670
ERR02680
ERR02690
ERR02700
ERR02710
ERR02720
ERR02730
ERR02740
ERR02750
ERR02760
ERR02770
ERR02780
ERR02790
ERR02800
ERR02810
ERR02820
ERR02830
ERR02840
ERR02850
ERR02860
ERR02870
ERR02880
ERR02890
ERR02900
ERR02910
ERR02920
ERR02930
ERR02940
ERR02950
ERR02960
ERR02970
ERR02980
ERR02990
ERR03000
ERR03010
ERR03020
ERR03030
ERR03040
ERR03050
ERR03060
ERR03070
ERR03080
ERR03090

```

700 IF(NR17-NPBC) 700,720,710
    WRITE(6,2000)
    WRITE(6,6000) IFILE
    ISTOP=1
    GO TO 720
710 WRITE(6,3000)
    WRITE(6,5000) IFILE
C 720 IF(LRB17-32) 730,750,740
730 WRITE(6,2000)
    WRITE(6,8000) IFILE
    ISTOP=1
    GO TO 750
740 WRITE(6,3000)
    WRITE(6,8500) IFILE
C
C      FILE NO 19
C
750 IFILE=19
    IF(NR19-NEL) 760,780,770
760 WRITE(6,2000)
    WRITE(6,6000) IFILE
    ISTOP=1
    GO TO 780
770 WRITE(6,3000)
    WRITE(6,5000) IFILE
C 780 IF(LRB19-480) 790,810,800
790 WRITE(6,2000)
    WRITE(6,8000) IFILE
    ISTOP=1
    GO TO 810
800 WRITE(6,3000)
    WRITE(6,8500) IFILE
C
C      FILE NO 20
C
810 IFILE=20
    IF(NR20-NN) 820,840,830
820 WRITE(6,2000)
    WRITE(6,6000) IFILE
    ISTOP=1
    GO TO 840
830 WRITE(6,3000)
    WRITE(6,5000) IFILE
C 840 IF(LPB20-NS*8) 850,870,860
850 WRITE(6,2000)
```



```

ERR031100
ERR031110
ERR031120
ERR031130
ERR031140
ERR031150
ERR031160
ERR031170
ERR031180
ERR031190
ERR032000
ERR032100
ERR032200
ERR032300
ERR032400
ERR032500
ERR032600
ERR032700
ERR032800
ERR032900
ERR033000
ERR033100
ERR033200
ERR033300
ERR033400
ERR033500
ERR033600
ERR033700
ERR033800
ERR033900
ERR034000
ERR034100
ERR034200
ERR034300
ERR034400
ERR034500
ERR034600
ERR034700
ERR034800
ERR034900
ERR035000
ERR035100
ERR035200
ERR035300
ERR035400
ERR035500
ERR035600
ERR035700

```

```

C      WRITE(6,8000) IFIL
      ISTOP=1
      GO TO 870
860    WRITE(6,3000)
      WRITE(6,8500) IFIL
C      IF(LRW20-NS*2) 880,900,890
870    WRITE(6,2000)
880    WRITE(6,9000) IFIL
      ISTOP=1
      GO TO 900
890    WRITE(6,3000)
      WRITE(6,9500) IFIL
900    IF((LPW20*4-LB20).EQ.0) GO TO 905
      WRITE(6,2000)
      WRITE(6,9700) IFIL
      ISTOP=1
C      FILE NO 21
C      IFIL=21
905    IF(NR21-NN*NS/NDF) 910,930,920
910    WRITE(6,2000)
      WRITE(6,6000) IFIL
      ISTOP=1
      GO TO 930
920    WRITE(6,3000)
      WRITE(6,5000) IFIL
C      IF(LPB21-24) 940,960,950
930    WRITE(6,2000)
940    WRITE(6,8000) IFIL
      ISTOP=1
      GO TO 960
950    WRITE(6,3000)
      WRITE(6,8500) IFIL
C      IF(LPW21-6) 970,990,980
960    WRITE(6,2000)
970    WRITE(6,9000) IFIL
      ISTOP=1
      GO TO 990
980    WRITE(6,3000)
      WRITE(6,9500) IFIL
990    IF((LPW21*4-LB21).EQ.0) GO TO 995
      WRITE(6,2000)
      WRITE(6,9700) IFIL
      ISTOP=1

```



```

C C C      FILE ND 22
          995 IF(NPBC2.EQ.0) GO TO 1050
              IF(ILE=22)
                IF(NP22-NPRC2) 1000,1020,1010
                  WRITE(6,2000)
                  IF(ILE
                    ISTOP=1
                    GO TO 1020
                  WRITE(6,3000)
                  IF(ILE
                    WRITE(6,5000)
                  IF(ILE
C          1020 IF(LRB22-24) 1030,1050,1040
          1030 WRITE(6,2000)
                  WRITE(6,8000)
                  IF(ILE
                    ISTOP=1
                    GO TO 1050
                  WRITE(6,3000)
                  IF(ILE
                    WRITE(6,8500)
                  IF(ILE
C          1040 NUMBER OF MATERIALS
C C C
C          1050 IF(NMAT.LE.10) GO TO 1055
                  WRITE(6,2000)
                  WRITE(6,7000)
                  ISTOP=1
C          1055 IF((ISTP9.EQ.0).AND.(ISTP16.EQ.0)) RETURN
                  WRITE(6,3000)
                  WRITE(6,9600)
                  WRITE(6,4000)
                  CALL SKIP
                  CALL MAIN1
C C C *****
C          ENTRY ERROR2
C C C
C          ENTRY 'ERROR2' CHECKS THE NUMBE OF RECORDS OF FILE NO 7
C C C
          IF(ILE=7)
            IF(NR7-MM>NN*NREC7) 1060,1080,1070
              WRITE(6,2000)
              IF(ILE
                ISTOP=1
                GO TO 1080
              WRITE(6,3000)
          1060
          1070

```


SKP00170
SKP00180

RETURN
END

SRT000020
SRT000030
SPT000040
SRT000050
SRT000060
SRT000070
SRT000080
SRT000090
SRT000100
SRT000110
SRT000120
SRT000130
SRT000140
SRT000150
SPT000160
SPT000170
SRT000180
SPT000190
SRT000200
SRT000210
SRT000220
SRT000230
SRT000240
SRT000250
SRT000260
SRT000270
SRT000280

SUBROUTINE SORT

IMPLICIT REAL*8(A-H,O-Z)
COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NN,MM,NS,NCOUNT,NST,NSTF,NCLD,
1 NPREC,NPRC2,NBCL,NBMAT,NCON21,NDFP,NLINE,NREC7,KEL
COMMON /NB2/NCON(20),NBC(4)
COMMON /B2/ ELAST(6,6),COORD(3),COREL(20,3),UJNT(3),SSSS1(7),
1 SSNN1(7)

IN THIS SUBROUTINE THE ARRAYS OF ELEMENT NODAL COORDINATES
, COREL(NPEL,NDF), ARE FORMED AND STORED ON DISK

REWIND 9
DO 300 I=1,NEL
READ(9) NCON
DO 200 J=1,NPEL
NTRACK=NCON(J)
CALL RDISKC(NTRACK)
DO 100 K=1,NDF
COREL(J,K)=COORD(K)
100 CONTINUE
200 CONTINUE
300 WRITE(19) COREL
CONTINUE
RETURN
END

SUBROUTINE MULT(A,B,C,NRA,NCA,NCB)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION A(1),B(1),C(1)
ZRO=0.000
DO 100 I=1,NRA

MLT000020
MLT000030
MLT000040
MLT000050
MLT000060

[illegible]

UUUUUU

```
SUBROUTINE SYMINV(A,N,B,C,IFLG)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION A(I),B(1),C(1)
```

[illegible]

```

IFLG=0
ZRQ=0.0D0
ARIGN=1.0D15
TRACE=ZRO
DO 5 I=1,N
IAD=(I-1)*N+1
IF(A(IAD).GE.ABIGN) GO TO 5
TRACE=TRACE+DABS(A(IAD))
CONTINUE
APZRO=TRACE*1.0D-12
IF(APZRO.GT.1.0D-20) APZRO=1.0D-20

```

```
TRACE=ZERO
DO 5 I=1,N
  IAD=(I-1)*N+I
  IF(A(IAD).GE.ABIGN) GO TO 5
  TRACE=TRACE+DABS(A(IAD))
```

```

5 CONTINUE
  APZRO=TRACE*1.0D-12
  IF (APZRC.GT.1.0D-20) APZRO=1.0D-20
  IP=IN

```

```
DO 10 I=2,LP
DC 10 J=1,LP
ICOL=J+LP*(I-2)
IROW=I+(J-1)*(P-1)
```

```

      IF(A(ICOL).EQ.A(IROW)) GO TO 10
      A(ICOL)=0.5D0*(A(ICOL)+A(IROW))
      A(IROW)=A(ICOL)
      9 CONTINUE

```

```
DO 25 I=1,N
  B(I)=ZERO
  II=(I-1)*N
```



```

DO 20 J=1,N
JJ=I+J
IF(A(JJ).GE.ABIGN) GO TO 20
B(I)=P(I)+DABS(A(JJ))
20 CONTINUE
25 ANR=ZRC
DO 30 I=1,N
ANR=DMAX1(ANR,B(I))
30 DO 145 J=1,N
NR=(I-1)*N
DO 100 J=1,N
K=NR+J
B(J)=A(K)
100 D=B(I)
IF(D.EQ.ZRO) GO TO 180
IF(DABS(D).LT.APZRO) IFLG=2
DO 110 J=1,N
C(J)=-B(J)/D
110 C
L=1
DO 130 J=1,N
M=L
DO 120 K=J,N
DR=DABS(B(J))
IF(DR.LE.1.0D-40) GO TO 115
DC=DABS(C(K))
IF(DC.LE.1.0D-40) GO TO 115
IF((DR.LE.APZRO).AND.(DC.LE.APZRO)) GO TO 115
A(L)=A(L)+B(J)*C(K)
115 CONTINUE
A(M)=A(L)
M=M+N
L=L+1
120 L=L+J
130 C
C(I)=-1.0D0/D
M=I
DO 140 J=1,N
K=NR+J
A(K)=C(J)
A(M)=C(J)
140 M=M+N
145 CONTINUE
C
NS=N*N
DO 150 J=1,NS

```


SMV00810
 SMV00820
 SMV00830
 SMV00840
 SMV00850
 SMV00860
 SMV00870
 SMV00880
 SMV00890
 SMV00900
 SMV00910
 SMV00920
 SMV00930
 SMV00940
 SMV00950
 SMV00960
 SMV00970
 SMV00980
 SMV00990
 SMV01000
 SMV01010
 SMV01020

```

C 150 A(J)=-A(J)
    DO 165 I=1,N
      B(I)=ZRO
      II=(I-I)*N
      DO 160 J=1,N
        JJ=II+J
        IF(A(JJ).EQ.1.0D0) GO TO 160
      B(I)=B(I)+DABS(A(JJ))
    CONTINUE
  160 CONTINUE
  165 AINR=ZRO
    DO 170 I=1,N
      AINR=DMAX1(AINR,B(I))
      IF(AINR.LT.1.0D-15) AINR=1.0D0/AINR
    CNBR=AINR*AINR
  2000 WRITE(6,2000) CNBR,AINR,AINR
      FORMAT(5X,'CONDITION NUMBER ',5X,1PD25.16,5X,2(1PD25.16))
  180 RETURN
      IFLG=1
      RETURN
      END
  
```


MAI000010
MAI000015
MAI000020
MAI000025
MAI000030

MAI 00 200
MAI 00 210

SMA000200
 SMA000300
 SMA000400
 SMA000500
 SMA000600
 SMA000700
 SMA000800

SMA02640

I	N	P	0	0	0	2	0
I	N	P	0	0	0	3	0
I	N	P	0	0	0	4	0
I	N	P	0	0	0	5	0
I	N	P	0	0	0	6	0

CONTROL PROGRAM

SAME AS * FOR QUADRATIC ELEMENTS *

STOP
END

SUBROUTINE MAIN

```

      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /NB1/ NEL, NDPRT, NPDEL, NDF, NEQ, NN, MM, NS, NCOUNT, NST, NSTF, NCLD,
      1 NPREC, NPBC2, NBCL, NBCH, NCMAT, NCON21, NDFE, NLINE, NREC7, KEL
      COMMON /B2/ ELAST (6,6), COORD (3), COPEL (32,3), UJNT (3), SSS1 (7),
      1 SSSN1 (7)
      *

```

* SAME AS FOR QUADRATIC ELEMENTS

D
N
M

SUBROUTINE INPUT

```

      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /NB1/NEL,NDPT,NDEL,NDF,NEQ,NIJ,MM,NS,NCOUNT,NST,NSTF,NCLD,
      1NPBC,NPBC2,NBCL,NBCH,NMAT,NCON21,NDF2,NLINE,NREC7,KEL

```


UUUUUUUUUUUUUUUUUUUU

```
COMMON /NB2/ NCON(32),NBC(4)
COMMON /B1/ ELDAT(10,3),CLOAD(4),BCON(3)
COMMON /B2/ ELAST(6,6),COORD(3),COREL(32,3),UJNT(3),SSSS1(7),
1SSNN1(7)
DIMENSION TITLE(10),FMT{10},CLOAD1(3),NCON1(16),NCON2(16)
EQUIVALENCE (CLOAD(1),CLOAD1(1)),(TITLE(1),COREL(1,1)),
1(FMT(1),COREL(1,3)),(NCON(1),NCON1(1)),(NCON(17),NCON2(1))
```

SAME AS FOR QUADRATIC ELEMENT

CONNECTIVITY MATRIX

```

READ(5,1000)      TITLE
WRITE(6,1100)     TITLE
WRREAD(6,1000)    TITLE
WRWRITE(6,1000)   TITLE
WRREAD(5,1000)    FMT
WRWRITE(6,1100)   FMT
NBRAND=0

```

SELECT THE APPROPRIATE SEGMENT (1) OR (2) DEPENDING ON WHETHER ONE OR MORE MATERIALS ARE PRESENT IN THE STRUCTURE

IF(NMAT.EQ.1) GO TO 150

(1) THIS SEGMENT IS USED FOR THE CASE OF MORE THAN ONE MATERIALS PRESENT IN THE STRUCTURE

```
DO 100 I=1,NELCON1  
  DDREAD(5,FMT),NCON1,NCON2,NCON21,  
    WRWRITE(6,FMT),NCON2,NCON21,  
    WRWRITE(9,FMT),NCON21
```

COMPUTATION OF HALF BANDWIDTH

```

NPELM=NPEL-1
DO 100 J=1,NPELM
  JP=J+1
  DO 100 K=JP,NPEL
    NBAND=MAXO(NBAND

```

```

1000 NBAND=MAXO(NBAND,IABS(NCON(J)-NCON(K)))
1001 DO 100 K=J,NFEL

```

uuu

UUUU UU

INP00500
INP00510
INP00520
INP00530
INP00540
INP00550
INP00555
INP00560
INP00565
INP00570
INP00580
INP00590
INP00600
INP00610
INP00620
INP00630
INP00640
INP00650
INP00660
INP00670
IMF00680
IMP00685
INP00690
INP00695
INP00700
INP00710
INP00720
INP00730
INP00740
INP00750
INP00760
INP00770
INP00780
INP00790

[illegible]


```

**
**
**
END
ELP00280
**
**
**
SUBROUTINE FSTF
IMPLICIT REAL*8(A-H,O-Z)
COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NN,MM,NS,NCOUNT,NST,NSTF,NCLD,
1INPRC,NPBC2,NBCH,NMAT,NCON21,NDEF,NLINE,NREC7,KEL
COMMON /B2/ ELAST(6,6),COORD(3),CORE.(32,3),UJNT(3),SSSS1(7),
1SSNN1(7)
COMMON /B3/ AK1(96,96),AK2(96,96),AK3(96,96),RB1(96),RB2(96),
1RB3(96)
DIMENSION STK(9216),AK(9216),B(576)
**
**
**
SAME AS FOR QUADRATIC ELEMENTS
**
**
**
END
FST00460
**
**
**
SUBROUTINE CUB
LISTINGS OF SUBROUTINES PERFORMING NUMERICAL INTEGRATION BY USE OF
TWO, THREE, FOUR, OR FIVE GAUSS POINTS ARE SHOWN IN APPENDIX
**
**
**

```


C*****C

SUBROUTINE FORMK(X,Y,Z,INDIC)

```

IMPLICIT REAL*8(A-H,O-Z)
COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NN,M,NS,NCOUNT,NST,NSTF,NCLD,
1NPEC,NPBC2,NBCL,NBMAT,NCON21,NDFP,NLINE,NREC7,KEL
COMMON /NB2/NCON(32),NBC(4),LM(32)
COMMON /B2/ ELAST(6,6),COORD(3),COREL(32,3)
COMMON /B3/ AK1(96,96),AK2(96,96),AK3(96,96),RB1(96),RB2(96),
1RB3(96)
DIMENSION AJ(3,3),AJIN(3,3),DNX(3,32),W1(3,32),B(6,96),B1(96,6),
2AK(96,96),CORDG(32,3),STK(96,96)
EQUIVALENCE (STK(1,1),AK1(1,1)),(AK(1,1),AK2(1,1)),(B(1,1),
1AK3(1,1)),(B1(1,1),AK3(1,7)),(W1(1,1),RB3(1))

```

```

DATA CORDG/1.0D0,0.3333333333333333,0.3333333333333333,-1.0D0,
1-1.0D0,-1.0D0,-1.0D0,-0.3333333333333333,0.3333333333333333,1.0D0,
21.0D0,1.0D0,1.0D0,-1.0D0,-1.0D0,1.0D0,1.0D0,-1.0D0,1.0D0,
31.0D0,0.3333333333333333,-0.3333333333333333,1.0D0,-1.0D0,-1.0D0,
4-1.0D0,-0.3333333333333333,0.3333333333333333,1.0D0,1.0D0,1.0D0,
51.0D0,1.0D0,1.0D0,1.0D0,0.3333333333333333,-0.3333333333333333,
6-1.0D0,-1.0D0,-1.0D0,-1.0D0,-0.3333333333333333,0.3333333333333333,
71.0D0,1.0D0,-1.0D0,-1.0D0,1.0D0,1.0D0,-1.0D0,1.0D0,1.0D0,
81.0D0,1.0D0,0.3333333333333333,-0.3333333333333333,-1.0D0,-1.0D0,
9-1.0D0,-1.0D0,-0.3333333333333333,0.3333333333333333,1.0D0,1.0D0,
A1.0D0,1.0D0,1.0D0,1.0D0,1.0D0,1.0D0,1.0D0,1.0D0,1.0D0,33333333
B0.3333333333333333,0.3333333333333333,0.3333333333333333,0.33333333
C3333333333,-0.3333333333333333,-0.3333333333333333,-0.333333333333
D3333,-0.3333333333333333,-1.0D0,-1.0D0,-1.0D0,-1.0D0,-1.0D0,-1.0D0
E,-1.0D0,-1.0D0,-1.0D0,-1.0D0,-1.0D0,-1.0D0,-1.0D0/

```

GENERAL FORMULAE FOR COMPUTATION OF DERIVATIVES OF SHAPE
FUNCTIONS WITH RESPECT TO XI, ETA, ZETA.

```

DFX(X,Y,Z,X1,Y1,Z1)=(1.0D0+Y*Y1)*(1.0D0+Z*Z1)*(9.0D0*X1*(X*X+Y*Y+
1Z*Z)-19.0D0*X1+18.0D0*X+18.0D0*X*X*Z1)/64.0D0
DX(X,Y,Z,X1,Y1,Z1)=9.0D0*(1.0D0+Y*Y1)*(1.0D0+Z*Z1)*(-2.0D0*X-27.0D
10*X*X*X1+9.0D0*X1)/64.0D0
DY(X,Z,X1,Y1,Z1)=9.0D0*(1.0D0-X*X)*(1.0D0+9.0D0*X*X1)*Y1*(1.0D0+
1Z*Z1)/64.0D0
DZ(X,Y,X1,Y1,Z1)=DY(X,Y,X1,Z1,Y1)

```

C*****C


```

COMPUTATION OF DERIVATIVES OF SHAPE FUNCTIONS WITH RESPECT
TO 'XI','ETA','ZETA' EVALUATED AT THE POINTS OF INTEGRATION
FMK00440
FMK00450
FMK00460
FMK00470
FMK00480
FMK00490
FMK00500
FMK00510
FMK00520
FMK00530
FMK00540
FMK00550
FMK00560
FMK00570
FMK00580
FMK00590
FMK00600
FMK00610
FMK00620
FMK00630
FMK00640
FMK00650
FMK00660
FMK00670
FMK00680
FMK00690
FMK00700
FMK00710
FMK00720
FMK00730
FMK00740
FMK00750
FMK00760
FMK00770
FMK00780
FMK00790
FMK00800
FMK00810
FMK00820
FMK00830
FMK00840
FMK00850
FMK00860
FMK00870
FMK00880
FMK00890
FMK00900
FMK00910
FMK00920
FMK00930

(1) CORNER NODES :      XI= +1 OR -1
                        ZETA= +1 OR -1
                        ETA= +1 OR -1
                        NODE # : 1,4,7,10,21,24,27,30

DO 100 I=1,2
  II=20*(I-1)+1
  IT=II+9
  DO 100 J=II,IT,3
    XI=CORDG(J,1)
    YI=CORDG(J,2)
    ZI=CORDG(J,3)
    W1(1,J)=DFX(X,Y,Z,XI,Y1,Z1)
    W1(2,J)=DFX(Y,X,Z,Y1,X1,Z1)
    W1(3,J)=DFX(Z,X,Y,Z1,X1,Y1)
  100

(2) MIDDLE NODES :      XI= + OR - 1/3
                        NODE # : 2,3,8,9,22,23,28,29

DO 200 K=1,2
  IK=20*(K-1)
  DO 200 I=1,2
    II=6*(I-1)+2+IK
    IT=II+1
    DO 200 J=II,IT
      XI=CORDG(J,1)
      YI=CORDG(J,2)
      ZI=CORDG(J,3)
      W1(1,J)=DX(X,Y,Z,XI,Y1,Z1)
      W1(2,J)=DY(X,Y,Z,X1,Y1,Z1)
      W1(3,J)=DZ(X,Y,Z,X1,Y1,Z1)
    200

(3) MIDDLE NODES :      ETA= + OR - 1/3
                        NODE # : 5,6,11,12,25,26,31,32

DO 300 K=1,2
  IK=20*(K-1)
  DO 300 I=1,2
    II=6*(I-1)+5+IK
    IT=II+1
    DO 300 J=II,IT
      XI=CORDG(J,1)
      YI=CORDG(J,2)
      ZI=CORDG(J,3)
  300

```


FMK00940
FMK00950
FMK00960
FMK00970
FMK00980
FMK00990
FMK01000
FMK01010
FMK01020
FMK01030
FMK01040
FMK01050
FMK01060
FMK01070
FMK01080
FMK01090
FMK01100
FMK01110

FMK01890
FMK01900

MRG00020
MRG00030
MRG00040
MRG00050
MRG00060
MRG00070
MRG00080
MRG00090
MRG00100

W1(1,J)=DY(Y, Z,Y1,X1,Z1)
W1(2,J)=DX(Y,X,Z,Y1,X1,Z1)
W1(3,J)=DZ(Y,X,Z,Y1,X1,Z1)

(4) MIDDLE NODES : ZETA= + OR - 1/3
NODE # : 13,14,15,16,17,18,19,20

DO 400 K=1,2
IK=4*(K-1)
II=13+IK
IT=II+3

DO 400 J=II,IT
X1=CORDG(J,1)
Y1=CORDG(J,2)
Z1=CORDG(J,3)
W1(1,J)=DZ(Z,Y,Z1,Y1,X1)
W1(2,J)=DY(Z,Y,X,Z1,Y1,X1)
W1(3,J)=DX(Z,Y,X,Z1,Y1,X1)

SAME AS FOR QUADRATIC ELEMENTS

RETURN
END

SUBROUTINE MERGE
IMPLICIT REAL*8(A-H,O-Z)
COMMON /NB1/NEL,NBPT,NPEL,NDF,NEQ,NN,NM,NS,NCOUNT,NST,NSTF,NCLD,
1 INPBC,NPBC2,NBCL,NBCH,NMAT,NCON21,NDFP,NLINE,NREC7,KEL
COMMON /NB2/ NCON(32),NBC(4)
COMMON /B3/ AK1(96,96),AK2(96,96),AK3(96,96),RB1(96),RB2(96),
1 RB3(96)
DIMENSION A(96,96),S(96,96),B(9216),T(9216),LM(32)

SAME AS FOR QUADRATIC ELEMENTS

[illegible]

[illegible]

SMV01010
SMV01020

*

*

RETURN
END

C

LISTING FOR AUXILIARY PROGRAM JCL

153

CCC

FILE NO 8

NREC8=4
IF(NPEL.EQ.32) NREC8=12
NRD1(2)=NEL*NREC8
LRWD1(2)=NSTF#2/NREC8
LRBD1(2)=LRWD1(2)*4

JCL00490
JCL00500
JCL00510
JCL00520
JCL00530
JCL00540
JCL00550
JCL00560
JCL00570
JCL00580
JCL00590
JCL00600
JCL00610
JCL00620
JCL00630
JCL00640
JCL00650
JCL00660
JCL00670
JCL00680
JCL00690
JCL00700
JCL00710
JCL00720
JCL00730
JCL00740
JCL00750
JCL00760
JCL00770
JCL00780
JCL00790
JCL00800
JCL00810
JCL00820
JCL00830
JCL00840
JCL00850
JCL00860
JCL00870
JCL00880
JCL00890
JCL00900
JCL00910
JCL00920
JCL00930
JCL00940
JCL00950
JCL00960

CCC

FILE NO 10

NRD2(1)=NDPT
LRWD2(1)=6
LRBD2(1)=LRWD2(1)*4

CCC

FILE NO 12

NRD2(2)=NDPT
LRWD2(2)=14
LRBD2(2)=LRWD2(2)*4

CCC

FILE NO 13

NRD2(3)=NDPT
LRWD2(3)=14
LRBD2(3)=LRWD2(3)*4

CCC

FILE NO 20

NRD2(4)=NN
LRWD2(4)=NS*2
LRBD2(4)=LRWD2(4)*4

CCC

FILE NO 21

NRD2(5)=NN*NS/NDF
LRWD2(5)=6
LRBD2(5)=24

CCCCC

(2) SEQUENTIAL FILES

FILE NO 9

NRD1(3)=NEL
LRBD1(3)=128
IF(NPEL.EQ.20) LRBD1(3)=80

C

FILES NO 14,15,16,17,19,22

```
NRS(1)=NCLD
NRS(2)=NPBC
NRS(3)=NEL
NRS(4)=NPBC
NRS(5)=NEL
NRS(6)=NPBC2
IF(NMAT.GT.1) GO TO 12
LRBS(3)=0
12 IF(NPBC2.GT.0) GO TO 15
LRBS(6)=0
```

CONTROL CARDS

```
15 WRITE(6,8000)
WRITE(7,8000)
WRITE(6,9600)
```

FILES NO 7,8,9

```
DO 30 I=1,3
NR=NRD1(I)
LRB=LRBD1(I)
CALL EXDIC(NR,4,0,0,IER,ANR,ALPHA)
CALL EXDIC(LRB,4,0,0,IER,ALRB,ALPHA)
REWIND 8
WRITE(8,2000) ANR
WRITE(8,2000) ALRB
REWIND 8
READ(8,3000) NRA
READ(8,3000) LRBA
DO 20 J=1,4
IF(NRA(J).EQ.IBL) NRA(J)=IZR
IF(LRBA(J).EQ.IBL) LRBA(J)=IZR
```

```
20 CONTINUE
WRITE(6,4000) NFD1(I),LRBA,NRA
WRITE(7,4100) NFD1(I),LRBA,NRA
30 CONTINUE
```

FILES NO 10,12,13,20,21

```
DO 50 I=1,5
NR=NRD2(I)
```

JCL00970
JCL00980
JCL00990
JCL01000
JCL01010
JCL01020
JCL01030
JCL01040
JCL01050
JCL01060
JCL01070
JCL01080
JCL01090
JCL01100
JCL01110
JCL01120
JCL01130
JCL01140
JCL01150
JCL01160
JCL01170
JCL01180
JCL01190
JCL01200
JCL01210
JCL01220
JCL01230
JCL01240
JCL01250
JCL01260
JCL01270
JCL01280
JCL01290
JCL01300
JCL01310
JCL01320
JCL01330
JCL01340
JCL01350
JCL01360
JCL01370
JCL01380
JCL01390
JCL01400
JCL01410
JCL01420
JCL01430
JCL01440

[illegible]

U

157

50

۵۵۰

55

U

C

2


```

1, I4,4X, 'NR12 =', I4/, ' ', I4,4X, 'LRB8 =', I4,4X, 'LRB10 =', I4,4X,
25X, 'LRB7 =', I4,4X, 'LRB8 =', I4,4X, 'LRW8 =', I4,18X, 'LRW10 =', I4,4X,
3, 'LRB12 =', I4/, ' ', I4,4X, 'NR7 =', I4,4X, ' ', I4,4X,
4, 'LRB12 =', I4/, ' ', I4,4X, 'NR7 =', I4,4X, ' ', I4,4X,
55X, 'NR13 =', I4,4X, 'NR14 =', I4,4X, 'NR15 =', I4,4X, 'NR16 =', I4,4X,
6, 'NR17 =', I4/, ' ', I4,4X, 'LRB14 =', I4,4X, 'LRB15 =', I4,4X,
75X, 'LRB13 =', I4,4X, 'LRB14 =', I4,4X, 'LRB15 =', I4,4X,
8, 'LRB17 =', I4/, ' ', I4,4X, 'NR20 =', I4,4X, 'NR21 =', I4,4X, 'NR22 =', I4,4X,
95X, 'LRW13 =', I4,4X, 'NR20 =', I4,4X, 'NR21 =', I4,4X, 'NR22 =', I4,4X,
A5X, 'NR19 =', I4,4X, 'LRB20 =', I4,4X, 'LRB21 =', I4,4X, 'LRB22 =', I4,4X,
B5X, 'LRB19 =', I4,4X, 'LRW21 =', I4,4X, ' ', I4,4X,
C19X, 'LRW20 =', I4,4X, 'NRX = NUMBER OF RECORDS OF FILE X', ' ',
9910 FORMAT(, ' ', I5X, 'NRX =', I4,4X, ' ', I4,4X, ' ', I4,4X, ' ', I4,4X,
25X, 'LRBX =', I4,4X, ' ', I4,4X, ' ', I4,4X, ' ', I4,4X, ' ', I4,4X,
3, 'LRWX =', I4,4X, ' ', I4,4X, ' ', I4,4X, ' ', I4,4X, ' ', I4,4X,
9920 FORMAT(, ' ', I5X, 'NOTE : IF DATA OF FILES NO 16 OR 22 ARE ZERO THESE
1, FILES ARE NOT REQUIRED, )
STOP
END

```

○○○○○

EDIC0670
EDIC0680
EDIC0690
EDIC0700
EDIC0710
EDIC0720
EDIC0730
EDIC0740
EDIC0750
EDIC0760
EDIC0770
EDIC0780
EDIC0790
EDIC0800
EDIC0810
EDIC0820
EDIC0830
EDIC0840
EDIC0850
EDIC0860
EDIC0870
EDIC0880
EDIC0890
EDIC0900
EDIC0910

[illegible]

Label	Operation	Address	Hex Data
SLOWLIM	DC	00000000	X'4040404060F9F9F9'
WORD	DC	00000001	X'4040404040404040'
IER	DC	00000002	X'00000000'
MAX1	DC	00000003	X'05F5E0FF'
MAX	DC	00000004	X'05F5E0FF'
MIN1	DC	00000005	X'FF676981'
MIN	DC	00000006	X'FF676981'
MAX2	DC	00000007	X'0000270F'
MIN2	DC	00000008	X'FFFFFC19'
NEXT	DC	00000009	6'4
	LTR	0000000A	JTISOK
	BZ	0000000B	6,*+4
	BCTR	0000000C	6'6
	BZ	0000000D	JTISOK
	LA	0000000E	0,2
	ST	0000000F	0,IER
	BC	00000010	15,RETURN
	LA	00000011	8,4
	CR	00000012	3,8
	RNE	00000013	EIGHTCAR
	MVC	00000014	MAX(4),MAX2
	MVC	00000015	MIN(4),MIN2
	MVC	00000016	LOWLIM(8),SLOWLIM
	L	00000017	0,MAX
	LR	00000018	7,2
	BC	00000019	13,NOTTOBIG
	MVC	0000001A	WORD(8),UPLIM
	LA	0000001B	0,1
	ST	0000001C	0,IER
	BC	0000001D	15,RETURN
	L	0000001E	0,MIN
	LR	0000001F	7,2
	BC	00000020	11,0
	MVC	00000021	11,INRANGE
	SR	00000022	WORD(8),LOWLIM
	BCTR	00000023	0,0
	BC	00000024	0,*+4
	BC	00000025	0,IER
	BC	00000026	15,RETURN
	MVI	00000027	7,2
	L	00000028	11,FLAG+1,X'01'
	BC	00000029	11,2
	MVI	0000002A	11,NONNEG
	BC	0000002B	11,FLAG+1,X'00'
	SR	0000002C	1,1
	BCTR	0000002D	1,*+4
	MR	0000002E	6,1
	LA	0000002F	8,4

LA	1,WORD+7
LR	11,1
LTR	0,10
BC	7,NOTZERO
MVC	WORD+7(1),ALPHA+1
BC	15,RETURN
SR	6,0
DR	6,0
AH	6,ALPHA
STH	6,TEMP-1
MVC	0(1,1),TEMP
BCT	1,#+4
LTR	7,7
BC	7,NOTZERO
LH	10,FLAG
BNZ	10,10
MVC	NOSIGN
BCT	0(1,1),MINUS
LR	1,#+4
LR	12,11
SR	12,1
SP	10,3
BZ	10,12
LTR	10,RETURN
BZ	4,4
LA	RTJSTFD
LR	1,1(0,1)
LR	14,10
MVC	11,12
MVC	0(1,14),0(1)
LA	0(1,1),BLANK
BC	1(1,0),1
BCT	14,1(0,14)
BC	11,SHIFTL
SR	1,#+4
BM	14,#+4
LTR	10,RETURN
BNP	1,RETURN
LR	11,14
LR	1,14
MVC	14,12
BCT	0(1,11),0(1)
	0(1,1),BLANK
	11,#+4

[illegible]

APPENDIX I

LISTINGS OF INTEGRATION SUBROUTINES

A. QUADRATIC ELEMENTS - TWO GAUSS POINTS

```

C SUBROUTINE CUB
C
C IMPLICIT REAL*8(A-H,O-Z)
C COMMON /NB1/NEL,NDPT,NPEL,NDE,NEQ,NN,MM,NS,NCOUNT,NST,NSTF,NCLD,
1NPBC,NPBC2,NBCL,NBCH,NMAT,NCON21,NDFP,NLINE,NREC7,KEL
C COMMON /B3/ AK1(60,60),AK2(60,60),AK3(60,60),RBI(60),RB2(60),
1RB3(60)
C DIMENSION STK(60,60),AK(60,60),B(6,60)
C EQUIVALENCE (STK(1,1),AK1(1,1)),(AK(1,1),AK2(1,1)),(B(1,1),
1AK3(1,1))
C DIMENSION XI(2)
C DATA XI/0.5773502691896258,-0.5773502691896258/
C
C DO 100 I=1,NST
C DO 100 J=1,NST
C STK(I,J)=0.0D0
C
C 100
C
C      SELECTION OF POINTS OF INTEGRATION AND FORMATION OF
C      INTEGRANT BY SUBROUTINE 'FORMK'
C
C DO 200 I=1,2
C X=XI(I)
C DO 200 J=1,2
C Y=XI(J)
C DO 200 K=1,2
C Z=XI(K)
C
C CALL FORMK(X,Y,Z,1)
C
C      INTEGRATION BY GAUSSIAN FORMULA AND FORMATION OF ELEMENT
C      STIFFNESS MATRIX
C
C DO 200 L=1,NST
C DO 200 M=L,NST
C STK(L,M)=STK(L,M)+ AK(L,M)
C
C 200
C DO 300 I=1,NST
C DO 300 J=I,NST
C STK(J,I)=STK(I,J)
C
C 300
C RETURN
C END

```


[illegible]

164

C. QUADRATIC ELEMENTS - FOUR GAUSS POINTS

```

SUBROUTINE CUB
IMPLICIT REAL*8(A-H,O-Z)
COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NN,MM,NS,NCCOUNT,NST,NSTF,NCLD,
1NPRC,NPRC2,NBCH,NMAT,NCON21,NDFP,NLINE,NRFC7,KEL
COMMON /B3/ AK1(60,60),AK2(60,60),AK3(60,60),RB1(60),RB2(60),
1RB3(60)
DIMENSION STK(60,60),AK(60,60),B(6,60),XI(4),AI(4),AIA(4,4,4)
EQUIVALENCE (STK(1,1),AK1(1,1)),(AK(1,1),AK2(1,1)),(B(1,1),
1AK3(1,1))
DATA XI/0.8611363115940526, 0.3399810435848563,
1-0.3399810435848563,-0.8611363115940526/
DATA AI/0.3478548451374539,0.6521451548625461,
100 C.6521451548625461,0.3478548451374539/
DO 100 I=1,NST
DO 100 J=1,NST
DO STK(I,J)=0.000
100 STK(I,J)=0.000
CCCC
COMPUTATION OF PRODUCTS OF WEIGHTS
DO 150 I=1,4
DO 150 J=1,4
DO 150 K=1,4
150 AIA(I,J,K)=AI(I)*AI(J)*AI(K)
CCCC
SELECTION OF POINTS OF INTEGRATION AND FORMATION OF
INTEGRANT BY SUBROUTINE 'FORMK'
DO 250 I=1,4
X=XI(I)
DO 250 J=1,4
Y=XI(J)
DO 250 K=1,4
Z=XI(K)
CALL FORMK(X,Y,Z,1)
CCCC
INTEGRATION BY GAUSSIAN FORMULA AND FORMATION OF ELEMENT
STIFFNESS MATRIX
DO 200 L=1,NST
DO 200 M=L,NST
STK(L,M)=STK(L,M)+AIA(I,J,K)*AK(L,M)
200 STK(L,M)=STK(L,M)+AIA(I,J,K)*AK(L,M)
CONTINUE
DO 300 I=1,NST
DO 300 J=I,NST
300 STK(J,I)=STK(I,J)
RETURN

```


D. CUBIC ELEMENTS - THREE GAUSS POINTS

```
CU30000020
CU30000040
CU30000050
CU30000060
CU30000070
CU30000080
CU30000090
CU30000100
CU30000110
CU30000120
CU30000130
CU30000150
CU30000160
CU30000170
CU30000180
CU30000190
CU30000200
CU30000210
CU30000220
CU30000230
CU30000240
CU30000250
CU30000260
CU30000270
CU30000280
CU30000290
CU30000300
CU30000310
CU30000320
CU30000330
CU30000340
CU30000350
CU30000360
CU30000370
CU30000380
CU30000390
CU30000400
CU30000410
CU30000420
CU30000430
CU30000440
CU30000450
CU30000460
CU30000470
CU30000480
CU30000490
CU30000500

SUBROUTINE CUB
  IMPLICIT REAL*8(A-H,C-Z)
  COMMON /NB1/NEL,NDPT,NBEL,NDE,NEQ,NN,MM,NS,NCOUNT,NST,NSTF,NCLD,
  INPRC,NPBC2,NRCL,NBCH,NMAT,NCON21,NDFP,NLINE,NREC7,KEL
  COMMON /B3/ AK1(96,96),AK2(96,96),AK3(96,96),RB1(96),RB2(96),
  1RB3(96)
  DIMENSION STK(96,96),AK(96,96),R(6,96),XI(3),AI(3),AIA(3,3,3)
  &EQUIVALENCE (STK(1,1),AK1(1,1)),(AK(1,1),AK2(1,1)),(B(1,1),
  1AK3(1,1))
  DATA XI/0.7745966692414834,0.0000,-0.7745966692414834/
  DATA AI/0.5555555555555556,0.8888888888888889,0.5555555555555556/
  DO 100 I=1,NST
  DO 100 J=1,NST
  100 STK(I,J)=0.0000

      COMPUTATION OF PRODUCTS OF WEIGHTS

      DO 150 I=1,3
      DO 150 J=1,3
      DO 150 K=1,3
      150 AIA(I,J,K)=AI(I)*AI(J)*AI(K)

      SELECTION OF POINTS OF INTEGRATION AND FORMATION OF
      INTEGRANT BY SUBROUTINE FORMK.

      DO 250 I=1,3
      X=XI(I)
      DO 250 J=1,3
      Y=XI(J)
      DO 250 K=1,3
      Z=XI(K)

      CALL FORMK(X,Y,Z,1)

      INTEGRATION BY GAUSSIAN FORMULA AND FORMATION OF ELEMENT
      STIFFNESS MATRIX

      DO 200 L=1,NST
      DO 200 M=L,NST
      STK(L,M)=STK(L,M)+AIA(I,J,K)*AK(L,M)
      200 CONTINUE
      DO 300 I=1,NST
      DO 300 J=I,NST
      300 STK(J,I)=STK(I,J)
      RETURN
      END
```


F. CUBIC ELEMENTS - FIVE GAUSS POINTS

```

SUBROUTINE CUB
  IMPLICIT REAL*8(A-H,O-Z)
  COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NN,MM,NS,NCCUNT,NST,NSTF,NCLD,
  INPBC,NPBC2,NBCL,NBCH,NMAT,NCON21,NDFP,NLINE,NREC7,KEL
  COMMON /B3/ AK1(96,96),AK2(96,96),AK3(96,96),RB1(96),RB2(96),
  1RB3(96)
  DIMENSION STK(96,96),AK(96,96),B(6,96)
  EQUIVALENCE (STK(1,1),AK1(1,1)),(AK(1,1),AK2(1,1)),(B(1,1),
  1AK3(1,1))
  DIMENSION XI(5),AIA(5,5,5)
  DATA XI/0.9061798459386640,0.5384693101056831,0.0D0,-0.53846931010
  156831,-0.9061798459386640/
  DATA AIA/I/O.2369268850561891,0.4786286704993665,0.5688888888888889,0
  1.4786286704993665,0.2369268850561891/
  DO 100 I=1,NST
  DO 100 J=1,NST
  100 STK(I,J)=0.0D0
  CCC

      COMPUTATION OF PRODUCTS OF WEIGHTS

      DO 150 I=1,5
      DO 150 J=1,5
      DO 150 K=1,5
      150 AIA(I,J,K)=AI(I)*AI(J)*AI(K)

      SELECTION OF POINTS OF INTEGRATION AND FORMATION OF
      INTEGRANT BY SUBROUTINE FORMK.

      DO 200 I=1,5
      X=XI(I)
      DO 200 J=1,5
      Y=XI(J)
      DO 200 K=1,5
      Z=XI(K)
      CALL FORMK(X,Y,Z,1)

      INTEGRATION BY GAUSSIAN FORMULA AND FORMATION OF ELEMENT
      STIFFNESS MATRIX

      DO 200 L=1,NST
      DO 200 M=L,NST
      STK(L,M)=STK(L,M)+AIA(I,J,K)*AK(L,M)
      200

      DO 300 I=1,NST
      DO 300 J=I,NST
      STK(J,I)=STK(I,J)
      300 K
      RETURN
  CCCC

```


LIST OF REFERENCES

1. Cantin, G., "Three Dimensional Finite Element Studies,"
(report in preparation).
2. Cantin, G., "An Equation Solver of Very Large Capacity,"
International Journal for Numerical Methods in Engi-
neering, v. 3, p. 379-388, 1971.
3. Fox, L., An Introduction to Numerical Linear Algebra,
Oxford University Press, 1965.
4. Holand, I. and others, Finite Element Methods in Stress
Analysis, Tapir, 1969.
5. Irons, B. M., "A Frontal Solution Program for Finite
Element Analysis," International Journal for Numerical
Methods in Engineering, v. 2, p. 5-32, 1970.
6. Przemieniecki, J. S., Theory of Matrix Structural Analysis,
McGraw-Hill, 1968.
7. Washizu, K., Variational Methods in Elasticity and
Plasticity, ch. 3, Pergamon Press, 1968.
8. Zienkewicz, O. C., The Finite Element Method in Engineering
Science, McGraw-Hill, 1971.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2. Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3. Naval Weapons Center c/o Dr. Arnold Adicoff China Lake, California	2
4. Professor Gilles Cantin, Code 59 Ci Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	5
5. Professor Robert E. Newton, Code 59 Ne Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
6. Asst Professor David Salinas, Code 59 Zc Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
7. Professor J. E. Brock, Code 59 Bc Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
8. Professor R. H. Nunn, Code 59 Nu Chairman, Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
9. LCDR Emmanuel Leonidas, HN Hellenic Navy Command Athens, Greece	2

	No. Copies
10. Professor O. C. Zienkewicz Department of Civil Engineering University of Wales Swansea, England	1
11. Professor T. H. Pian Massachusetts Institute of Technology Boston, Massachusetts	1
12. Professor Ray W. Clough University of California Berkeley, California	1
13. Professor Edward L. Wilson University of California Berkeley, California	1
14. Professor David N. Murray University of Alberta Edmonton, Alberta, Canada	1
15. LT C. Pfeifer, USN Naval Destroyer School U.S. Naval Base Newport, Rhode Island	1

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

ORIGINATING ACTIVITY (Corporate author)	2a. REPORT SECURITY CLASSIFICATION
	2b. GROUP

Naval Postgraduate School
Monterey, California 93940

Unclassified

REPORT TITLE

A GENERAL PURPOSE THREE DIMENSIONAL
STRESS ANALYSIS PROGRAM

DESCRIPTIVE NOTES (Type of report and, inclusive dates)

Master's Thesis; December 1971

AUTHOR(S) (First name, middle initial, last name)

Emmanuel Leonidas
Lieutenant Commander, Hellenic Navy

REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
December 1971	173	8
8. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)	
9. PROJECT NO.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	

10. DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
-------------------------	----------------------------------

Naval Postgraduate School
Monterey, California 93940

13. ABSTRACT

A computerized solution for three dimensional stress analysis was developed. The solution was based on the finite element technique employing twenty and thirty-two nodal point three dimensional isoparametric elements. The solution is applicable to linear elastic structures composed of homogeneous and isotropic materials subjected to static loading under constant temperature. Deformed states are restricted to small displacements and displacement gradients. The program is coded in standard FORTRAN IV and is machine independent except for one subroutine which may be adjusted to specific installations. Independence of core space requirements is achieved by use of direct access storage facilities.

KEY WORDS

LINK A

LINK B

LINK C

ROLE

WT

ROLE

WT

ROLE

WT

finite element

three dimensional stress analysis

16 APR 73

DINDERY

20901

Thesis

133154

L543 Leonidas

c.1

A general purpose
three dimensional str-
ess analysis program

16 APR 73

DINDERY
20901

Thesis

133154

L543 Leonidas

c.1

A general purpose
three dimensional str-
ess analysis program.

thesL543

A general purpose three dimensional stre



3 2768 002 11839 0
DUDLEY KNOX LIBRARY